



Deliverable D4.8

GitHub Repository of EDIAQI Models Code - Version 2

Work Package 4

MONITOR – Pilots, data and campaigns

Version: Final



This project has received funding from the European Union's
Horizon Europe Framework Programme under grant agreement N° 101057497

Deliverable Overview

This deliverable provides an overview of the open-source EDIAQI GitHub repository that contains code used within the project. The EDIAQI GitHub repository will be continuously enhanced by partners throughout the project lifecycle.

Additional Information

Type: Other

Dissemination Level: PU – Public

Official Submission Date: 31st of July 2025

Actual Submission Date: 29th of July 2025



Disclaimer

This document contains material, which is the copyright of certain EDIAQI partners, and may not be reproduced or copied without permission. All EDIAQI consortium partners have agreed to the full publication of this document if not declared “Confidential”. The commercial use of any information contained in this document may require a license from the proprietor of that information. The reproduction of this document or of parts of it requires an agreement with the proprietor of that information.

Document Revision History

Version	Date	Description	Partners
V0.1	22nd May 2025	First draft of deliverable V2.0	KNOW
V0.2	21 st July 2025	Internal review of the deliverable	FTMC
V1.0	25 th July 2025	Update after the internal review	KNOW
FINAL	28 th of July 2025	Final quality check	LC

Authors and Reviewers

Authors

- Kristina Časni (KNOW)
- Heimo Gursch (KNOW)
- Lorenz Dirry (KNOW)

Reviewers

- Ivana Banić (SCH)
- Steigvilė Byčenkienė (FTMC)

Statement of Originality

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.



Table of Contents

Executive Summary	6
1. Introduction	7
2. Consortium Overview and Contributions.....	8
3. GitHub Structure	11
3.1. Data Platform - Task 4.1	11
3.2. Indoor Air Pollution Observation	13
3.3. Digital Twins.....	13
3.4. Mathematical Models and AI.....	14
3.5. Health Data Analysis.....	14
3.6. Interoperability Management	15
3.7. Microbiome Analysis	16
3.8. Collocation Study	16
4. Additions and Modifications since D4.5	17
4.1. Newly Introduced Repositories.....	17
4.2. Modifications to Existing Repositories	17
5. Submission and Review Process	18
6. Conclusion.....	19
Annex I – User Guide: KNOW-Center Data Ecosystem	20



List of Figures

Figure 1. Repositories of the EDIAQI GitHub.	8
Figure 2. Screenshot of the EDIAQI profile on GitHub.....	11



Executive Summary

The aim of this deliverable is to provide an overview of the EDIAQI GitHub page under the GitHub name EDIAQI and its structure. The page itself can be accessed by following the link below. In the deliverable we provide information on the structure and what visitors can expect to see in each of the repositories and sub-repositories. The EDIAQI GitHub will form the central access point for open-source model code that is and will be the outcome from the project. The repository will be continually enhanced and expanded as the project advances.

[Access the EDIAQI project's GitHub](#)



1. Introduction

The EDIAQI project intends to share as many details as possible of the executed work within the project, following open science practices to make research results accessible and reusable for both scientists and the public. One of the practices implemented is publishing work as pre-prints and in open access journals. Research data generated within the project will be accessible through IPCHEM. Where data sharing is not feasible due to GDPR compliance, ethical considerations, commercial sensitivity or any other reason, such restrictions will be transparently documented in the accompanying metadata. To support transparency and long-term usability the project's research data will be stored on the KNOW Data Platform (as described in Deliverable D4.1), making both data and metadata publicly accessible. In addition, project's code is made available and well documented via GitHub, forming a key component of the project's open science approach. This strategy is designed to encourage continued research by external experts and facilitate ongoing collaboration among partners, thereby advancing knowledge on indoor air quality and its health implications.



2. Consortium Overview and Contributions

The EDIAQI consortium is comprised of eighteen partners, with responsibilities distributed across seven closely interconnected work packages. The project's publicly available code is hosted on the EDIAQI GitHub profile and structured into eight repositories, each reflecting a key area of work within the project, as shown in Figure 1.

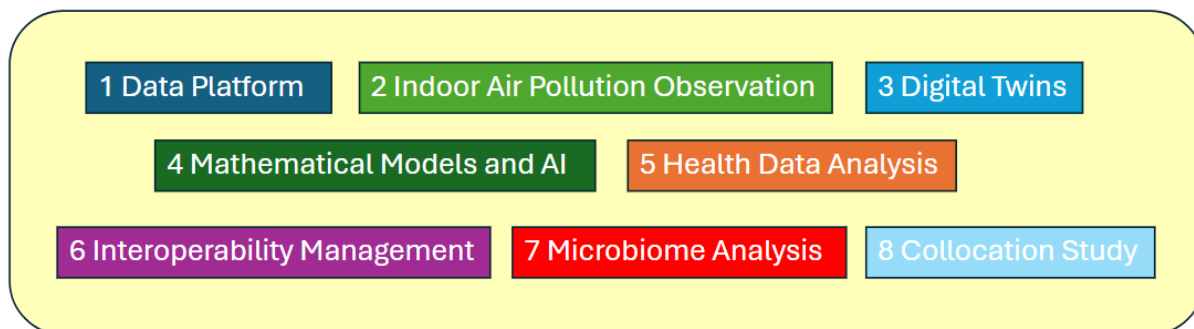


Figure 1. Repositories of the EDIAQI GitHub.

Of the five main repositories, four are associated with Work Package 4 (WP4): Data Platform, Interoperability Management, Digital Twins, and Mathematical Models and AI. The structure of these repositories mirrors the internal organisation of the EDIAQI project, with each repository corresponding to a dedicated functional domain.

- **Data Platform - Task 4.1** focuses on the development of the project central Data Platform. This task is led by the KNOW partner, in collaboration with LC, ASC, THIN, DEDA, TalTech, and LAS. The corresponding GitHub repository contains the code necessary for platform development and implementation within the project. The Data Platform was designed to support data-driven collaboration across the consortium, ensuring FAIR data management, secure data sharing, and reproducible analytics.
- **Indoor Air Pollution Observation** is part of Work Package 3 (WP3) and is led by TROPOS, with contributions from IMROH, ANT, TUG, UMOL, TalTech, LAS, and FTMC. It includes the Air Quality Sensor Evaluation Toolkit (Task 3.1), which aims to identify the most suitable approaches, tools and methods for characterising indoor air pollution and to estimating concentration levels. These evaluations are conducted using low-cost sensors tested for (correlation) at TROPOS laboratories in parallel to real-life experiments. This main repository also contains a sub-repository title



Microbiome, developed by TUG, which contains basic data analysis workflow for 16S rRNA gene fragment and ITS region amplicon sequencing. More details and the associated code are available in the respective GitHub sub-repository.

- **Digital Twins** – It is intended to host the code developed under Task 4.4, titled Hybrid approach for digital twins, where ASC is building physics-based models using data from the pilots, collaborating with partners from TROPOS and KNOW. The goal of this task is to create simulated environments that are an adequate representation of the physical world to better understand indoor air quality dynamics.
- **Mathematical Models and AI** - the final component of WP4 corresponds to Task 4.5 and is led by KNOW, in collaboration with ASC, WINGS, TROPOS, TalTech, LAS, and NIB. The Task focuses on applying data analysis and machine learning techniques to project' dataset. As part of this task, the Simulation Toolbox has been developed by RegionH and KNOW to provide a public tool for risk assessment given location, surroundings, building type, and information on inhabitants' information. Another key output is the application of machine learning to improve the performance of low-cost air quality sensors, developed by ASC. Detailed information and code are available in the corresponding GitHub sub-repository.
- **Health Data Analysis** – This repository contains code developed under Work Package 5 (WP5), which aims to assess both the short- and long-term effects of indoor air pollution on human health. The work is structured across three tasks: Task 5.1 involves the analysis of retrospective cohorts, led by RegionH and supported by KNOW, TUG, and SCH). Task 5.2 focuses on prospective cohorts' analysis, led by SCH with contributions from KNOW, IMROH, ASC, WINGS, RegionH, ANT, TUG, and TROPOS. Task 5.3 addresses toxicological studies, conducted by IMROH as the lead partner with contributions from RegionH, UMOL, and NIB. As of the submission date, the repository includes analyses of the ATOPICA and COPSAC2000 cohorts.
- **Interoperability Management** – The component is part of Work Package 4 (WP4) and is led by DEDA, with contributions from WINGS, KNOW, ASC, LAS, and TalTech. The task focuses on developing a unified interoperability approach and a corresponding technical solution to facilitate access to relevant data sources,



including those aligned with FAIR data. A central element of this effort is the integration and use of the Data Platform developed under Task 4.1 established at KNOW.

- **Microbiome Analysis** – This Task is a key component of the Zagreb Pilot (P3) and is led by TUG, with support from ANT, IMROH, and SCH. It involves the analysis of house dust samples collected from the participants in the SCH2021 cohort (home environment) to characterise the composition of the indoor microbiome.
- **Collocation Study** - This study is a central part of the Evaluation of low-cost sensorics campaign (C1), led by TROPOS and supported by IMROH. This study involves a comprehensive performance evaluation of various consumer-grade, low-cost sensors by comparing their measurements against scientific reference instrumentation within a controlled laboratory chamber.

More information about the project, its consortium, and work packages can be found on the [EDIAQI Project Website](#).



3. GitHub Structure

The EDIAQI project's GitHub page currently hosts the main repositories listed from the previous section, as illustrated in Figure 2. Each repository contains open-source code corresponding to the specific tasks within the project, aligned with the repository's name and scope. For enhanced clarity and user guidance, each sub-repository includes a README.md files. The text from each of these files is displayed to the users by default when they navigate to an individual sub-repository to detail the work completed within the corresponding task, how the current code is organised, and programming languages used for code development (e.g. Python, R, etc.).

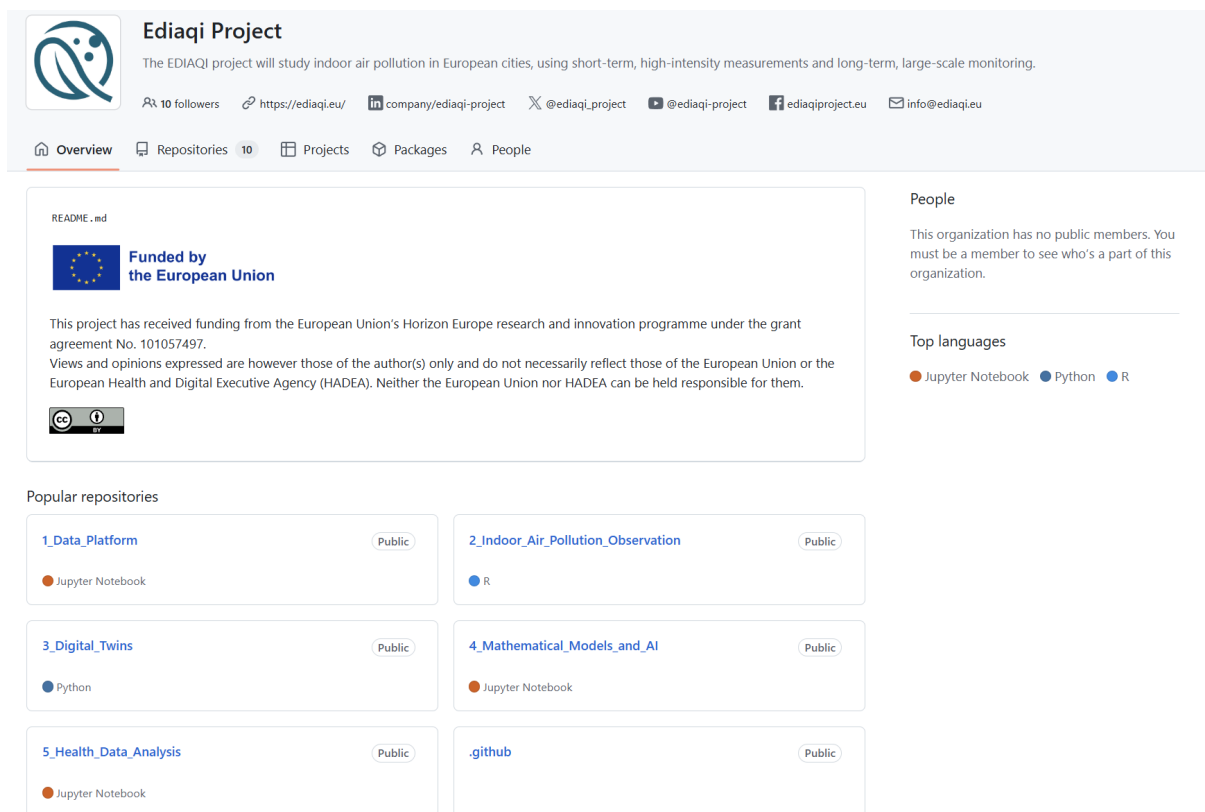


Figure 2. Screenshot of the EDIAQI profile on GitHub

The content structure of each repository is following outlined:

3.1. Data Platform - Task 4.1

- **01. Main page**



This project has received funding from the European Union's Horizon Europe Framework Programme under grant agreement N° 101057497

The landing hub of the project that provides centralised access to all tools, applications, and websites used across the EDIAQI project.

- **02. Data Monitoring**

Interface to explore and monitor data stored on the FROST-Server. Authenticated users can check for data integrity, including duplicate entries per pilot or project.

- **03. FROST-Server**

Core server implementing the OGC SensorThings API to manage and serve IoT-based observation data. It supports real-time querying, ingestion, and publication of sensor measurements.

- **04. Publish FROST to Data Platform**

Tool for exporting selected observation data from the FROST-Server into .csv format, with direct upload capability to the Data Platform where a DOI is generated for each dataset.

- **05. Jupyterhub**

Analytical environment where users can perform reproducible research in Python or R using Jupyter Notebooks. Integration with iRODS allows direct access to stored data and metadata.

- **06. Data Platform**

Data management platform based on iRODS with the MetaLNx web interface, used to store, share, and publish project-related datasets both internally and externally.

- **07. Keycloak**

Identity and access management tool used for centralised login and user management across all project services—except the iRODS platform.

- **08. CSV UI Upload**

Web interface that allows users to upload .csv files into the FROST-Server by mapping spreadsheet columns to SensorThings entities in a guided, user-friendly manner.



3.2. Indoor Air Pollution Observation

- **Microbiome**

This folder contains a structured R-based workflow for analysing microbiome sequencing data, specifically 16S rRNA and ITS region amplicons. The analysis includes relative and differential abundance, alpha and beta diversity, core microbiome identification, and linear modelling. It assumes input data such as ASV/OTU tables, taxonomic classifications, and sample metadata. The provided scripts support fundamental ecological and statistical analyses of microbial communities derived from sequencing experiments.

- **Air Quality Sensor Evaluation**

This toolkit supports the performance evaluation of air quality sensors using a comprehensive suite of R scripts. It includes methods for assessing accuracy, precision, correlation, and linear model behaviour of sensors, based on reference measurements and standardised metrics. The structure is modular, with functions for data retrieval from both internal and external sources (e.g., UBA API, cloud services) and is designed to work within a specific data architecture. Plotting, labelling, and evaluation are automated but require consistent naming and folder setup, with use cases anchored in validated scientific protocols.

3.3. Digital Twins

- **Model_Training**

This folder contains the full pipeline for developing predictive models used in Task 4.4 of the project. It includes scripts for data preprocessing, model training, and utility functions. The setup is modular and configurable, with parameters and dependencies clearly defined in separate configuration files. The included README.md provides step-by-step instructions for executing the training process, ensuring reproducibility.

- **Digital_Twin**

This folder implements the backend of the Digital Twin application for simulating



indoor air quality. It integrates trained models with real-time user input, applying preprocessing and inference to simulate environmental conditions. The entire application is containerised using Docker and Docker Compose, making it portable and easy to deploy across environments. The backend is structured for maintainability and documented for developers in its own README.md file.

3.4. Mathematical Models and AI

- **Simulation_Toolbox**

The Simulation Toolbox is an interactive web-based application developed using Streamlit and Docker. It allows users to estimate indoor air quality indicators such as PM2.5 and NO2 concentrations based on household and lifestyle inputs. The tool features an intuitive interface with sliders, checkboxes, and input fields organised into clear sections, and it requires a valid address in Copenhagen to function. Hosted online, the tool requires no installation, making it accessible and user-friendly for a wide audience.

[Access the Toolbox](#)

- **ML_for_Low-Cost_Sensors**

This folder includes a Jupyter notebook designed to enhance the accuracy of low-cost air quality sensors using machine learning. By comparing data from low-cost and reference-grade sensors, the notebook trains a random forest regression model to improve prediction accuracy. It supports hyperparameter optimisation and outputs all results to a dedicated folder.

3.5. Health Data Analysis

- **ATOPICA_cohort**

This module contains code for the analysis of the ATOPICA cohort, comprising comprehensive health and environmental data from over 4,000 children aged 3–11 across three Croatian regions with varying environmental exposures. The analysis explores associations between demographic, environmental, and socio-economic



factors and health outcomes using statistical methods such as univariate testing, logistic regression, and machine learning. A Random Forest Classifier with 5-fold cross-validation and hyperparameter tuning via Optuna is used to enhance predictive accuracy. This work was conducted as part of the EDIAQI project and is suitable for public dissemination.

- **COPSAC2000_cohort**

This component documents the simulation and modelling workflow for Danish households, based on the COPSAC2000 clinical cohort. It integrates synthetic PM_{2.5} and NO₂ data, indoor pollution sources and sinks, and external features obtained from the Danish Building Registry (BBR), OpenStreetMap, and DMI climate APIs. The methodology includes spatial and temporal integration of environmental variables, along with predictive modelling using training scripts focused on PM_{2.5} and NO₂ outcomes. Although still in an early development phase, the code represents an enhanced version of the simulation tool initially delivered in EDIAQI-5.1 and is intended for further refinement in an upcoming journal publication.

3.6. Interoperability Management

- **FROST_StaticData**

This script facilitates the creation of static SensorThings API (STA) entities by generating JSON payloads based on Excel input files. Designed for project pilots such as Seville and Vilnius, it supports creating core STA components like Things, Locations, Features of Interest, Observed Properties, and Datastreams. Each pilot may require geometry-specific handling (e.g., MultiLineString for Seville, Point/Polygon for Vilnius), and the output can be customised in terms of file names, IDs, and counters. The scripts are executed within a Jupyter Notebook, requiring users to follow a defined cell sequence and configure file paths accordingly.

- **LAS_FROST_DynamicData**

This module contains scripts for dynamically ingesting real-time and historical sensor data into the Lab Service FROST server using AWS APIs. Logging of operations is handled through CSV logs that track each execution. Additional scripts support direct



API queries and batch historical data ingestion using customisable time ranges and Netpid sensor IDs.

3.7. Microbiome Analysis

This project demonstrates a microbiome data analysis workflow in R using the phyloseq package with simulated data. It includes alpha diversity (Shannon, Simpson), beta diversity (PCoA with Bray-Curtis), and relative abundance visualisations.

3.8. Collocation Study

This repository contains code and datasets for a comprehensive sensor collocation study examining indoor air quality and environmental parameters across various controlled experiments. It involves collecting and analysing data from multiple sensors placed in indoor and outdoor environments to compare their performance and understanding air quality dynamics during various household activities and scenarios.



4. Additions and Modifications since D4.5

Since the release of Deliverable D4.5, several key additions and modifications have been made across the repositories. These changes reflect both the integration of new functionalities and the refinement of existing workflows to support improved interoperability, data analysis, and digital twin capabilities.

4.1. Newly Introduced Repositories

As already described in previous chapters the following three repositories have been added since the last version of this deliverable:

- Digital Twin
- Interoperability Management
- Microbiome Analysis
- Collocation Study

4.2. Modifications to Existing Repositories

In addition to the new repositories, several modifications were made to existing components:

- In the Indoor_Air_Pollution_Observation repository, a new function, `getData_VaisalaCO2`, was added to a data retrieval script. This enables direct access to CO₂ measurements from Vaisala sensors, enhancing the system's compatibility with commercial air quality hardware.
- In the Data Platform Repository, a new synchronisation script was developed specifically for the Vilnius campaign, which allows for automated synchronisation of data across FROST-Servers.



5. Submission and Review Process

Some of the eighteen partners who are part of the EDIAQI project contribute with different analyses, tools and developed models, resulting in code that can be publicly shared via GitHub. The EDIAQI GitHub collects all the code from the EDIAQI project that can be publicly shared.

Only approved members of the EDIAQI GitHub can submit their work, and the process is as follows:

- Partners develop code, models, tools, etc.
- Partners have internal development guidelines for quality assurance.
- Code is shared within the EDIAQI GitHub, but it is not available to the public during code reviewing and testing:
 - The review is done from partners in the same work package, who are familiar with the requirements, but did not work on the code themselves.
 - The partners must agree that the code fulfils the EDIAQI guidelines before it is published. This process is inspired by the review process of the deliverables.
- Code is made available to the public via GitHub once it is finally tested and reviewed.

API keys and other confidential information, such as passwords, should not be shared via GitHub. To address this, the solution in EDIAQI is to create a separate file that holds all confidential information of the work, typically as a YAML or properties file. The sensitive information is then removed from the file, so that only the names of the sensitive information remain inside, and the file can then be submitted to the GitHub. In the README.md file the content of the sensitive file should be explained, so that users can easily insert there their own API keys, passwords, etc.

During the review process, the partners check if the handling of sensitive information was done correctly.



6. Conclusion

The GitHub repository contains the open-source code generated during the project's duration, structured to be intuitive for external users who were not part of the EDIAQI project. This document has detailed the significant evolution of the repository since the previous report (D4.5), reflecting the project's progress. The repository is now structured into eight main repositories that intuitively mirror the project's core work packages and tasks, making the code accessible to both consortium members and the wider scientific community.

Key developments include the introduction of four new repositories for the Digital Twin, Interoperability Management, Microbiome Analysis, and the Collocation Study, alongside developments to existing repositories. These additions provide new tools and refined workflows for data analysis, simulation, and platform interoperability. A robust submission and review process remains in place to ensure all publicly shared code is of high quality, well-documented, and securely managed.



Annex I – User Guide: KNOW-Center Data Ecosystem

For a comprehensive guide on utilising the KNOW-Center Data Ecosystem, please refer to the "**KDP Usage Guide**" appendix below. This ecosystem provides a centralised platform for managing, sharing, and analysing research data in compliance with the FAIR principles. The guide describes interaction with the central storage systems: iRODS for file-based data and the FROST-Server for time-series sensor data. It also details data analysis within the collaborative JupyterHub environment and describes various data transfer methods. Following these guidelines ensures that data is handled correctly to promote reproducible and collaborative research.





Usage Guide

KNOW-Center Data Ecosystem

Central Data Access Point
Version: 1.0



This project has received funding from the European Union's
Horizon Europe Framework Programme under grant agreement N° 101057497.

1. Address list of services and contact for access	24
2. Introduction to the Know-Center Data Ecosystem	25
2.1. EDIAQI Data Flow	26
2.2. Base Platform Architecture	28
Data Discovery & Management	28
Data Analytics	29
2.3. Data Integration	30
Transfer data to KDP	30
Setup own iRODS zone	31
3. FAIR principles and purpose of the KNOW-Center Data Platform.....	32
3.1. Findability	32
3.2. Accessibility.....	32
3.3. Data Interoperability	33
3.4. Reusability	33
4. Usage of the KNOW-Center Data Platform.....	34
4.1. Access platform & Login	34
Metalnx.....	34
iCommands	34
4.2. Upload, Download, Rename & Delete File(s)	35
Metalnx.....	35
iCommands	36
4.3. Add Folder, Move & Copy File(s).....	37
Metalnx.....	37
iCommands	38
4.4. Create Templates & Add Metadata.....	38
Metalnx.....	38
iCommands	40
4.5. Permissions & Shared Links	40



Metalnx.....	40
iCommands	41
4.6. Jupyterhub	42
Supported Kernels	42
Example Notebooks.....	42
Additional Resources.....	43
5. Usage of Data Support Interoperability Tools	44
5.1. CSV Upload GUI	44
Login and Select a Thing	44
Upload CSV File.....	44
Mapping Value Columns with Datastreams	44
Data duplicates and finalizing Upload	45
Additional Options	45
5.2. Frost iRODS Publish Bridge	45
Filter by Thing, Sensor, and Datastream.....	45
Select a Date Range	46
Preview the Data	46
Publish to iRODS	46
5.3. Data Monitoring	47
Overall Monitoring	47
Pilot Monitoring	48
Check Duplicates.....	49
View Data in Details	50
Check Begin and End time	50
5.4. Data Troubleshooting	50
AQ Data Comparison	52



1. Address list of services and contact for access

Overview dashboard: <https://kdp-ediaqi.know-center.at>

Web File storage system Metalnx: <https://kdp-ediaqi.know-center.at/metalnx>

Jupyterhub: <https://kdp-ediaqi-jupyterhub.know-center.at>

Central KNOW-FROST-Server: <https://kdp-ediaqi.know-center.at/FROST-Server>

CSV Upload GUI: <https://kdp-ediaqi-upload.know-center.at>

Frost iRODS Bridge: <https://kdp-ediaqi-publish.know-center.at>

Data Monitoring: <https://kdp-ediaqi-data-monitoring.know-center.at>

Keycloak: <https://kdp-ediaqi.know-center.at/keycloak>

Data Troubleshooting and AQ Data Comparison: <https://ediaqi-dashboard.know-center.at/>

Accounts and access to the KNOW Center Data ecosystem is provided through

Lorenz Dirry: ldirry@know-center.at

Heimo Gursch: hgursch@know-center.at



2. Introduction to the Know-Center Data Ecosystem

This chapter introduces the user to the provided Know-Center services and describes the design and guidelines for the Know-Center data platform (KDP). It is structured into three main sections:

- EDIAQI Data Flow
- Base Platform Architecture
- Data Integration

The KDP is designed to provide a secure, scalable, and robust system for managing and sharing data. To do this, the KDP consists of different components, each of them designed for a specific task:

- iRODS¹ data management system for distributed and replicated storage of file-based data.
- FROST-Server² is incorporated into the data ecosystem to enable efficient management, retrieval and analysis of time-series sensor data in compliance with OGC SensorThings API standards³.
- JupyterHub⁴ is integrated within the KDP to provide researchers with a collaborative environment for interactive computing, supporting reproducible workflows through shared, executable notebooks in the programming languages Python and R. JupyterHub is connected to iRODS and the FROST server, so that all data is iRODS and FROST can be accessed from JupyterHub.
- GUI to publish data from FROST to iRODS is designed to select data from the FROST server, compose this data into a CSV file and put it in iRODS. This is needed so that the data can be made available to others, for example when used in a publication.

¹ <https://irods.org/>

² <https://www.iosb.fraunhofer.de/en/projects-and-products/frost-server.html>

³ <https://www.ogc.org/standard/sensorthings/>

⁴ <https://jupyterhub.readthedocs.io/en/stable/>



While these are the main components that are intended to for the data management and data analysis and are therefore relevant to most users. However, there are also other components of the KDP that are more focused and therefore not as relevant to all users:

- CSV Upload UI was developed to allow to upload measurement data in the form of CSV files from devices that cannot stream their data to the FROST server
- Data Monitoring is a dedicated application developed to give the data providers an overview about the data quality and volume of the data they are delivering to the FROST server
- Data Checker implemented as Apache Airflow⁵ workflows automates data quality checks to see if there are missing data, duplicates in the data or sensors deliver constant values over long times.
- Grafana⁶ allows to create visualisations from the data in the FROST server and make these visualisations available to an arbitrary user group
- Ther are further applications, like Keycloak⁷ for the user management and others, but users are not interacting with these applications and only admins do. Hence, they are not described here in detail.

The project team at Know-Center developed the KDP with the FAIR principles in mind and a strong focus on reproducible research.

2.1. EDIAQI Data Flow

The data flow begins with the input from several pilot sites and proceeds through a series of processes and components that make up the KDP in EDIAQI. The components and the data flow are show in [Figure 3](#).

The core of the KDP consists of the FROST-Server and iRODS. The FROST-Server plays a critical role in the EDIAQI data flow by managing and storing time-series data within a

⁵ <https://airflow.apache.org/>

⁶ <https://grafana.com/>

⁷ <https://www.keycloak.org/>

PostgreSQL database. This server is designed to handle sensor data efficiently, following the OGC SensorThings API standards. In parallel, iRODS serves as the central storage system for file-based data that guarantees stability and scalability.

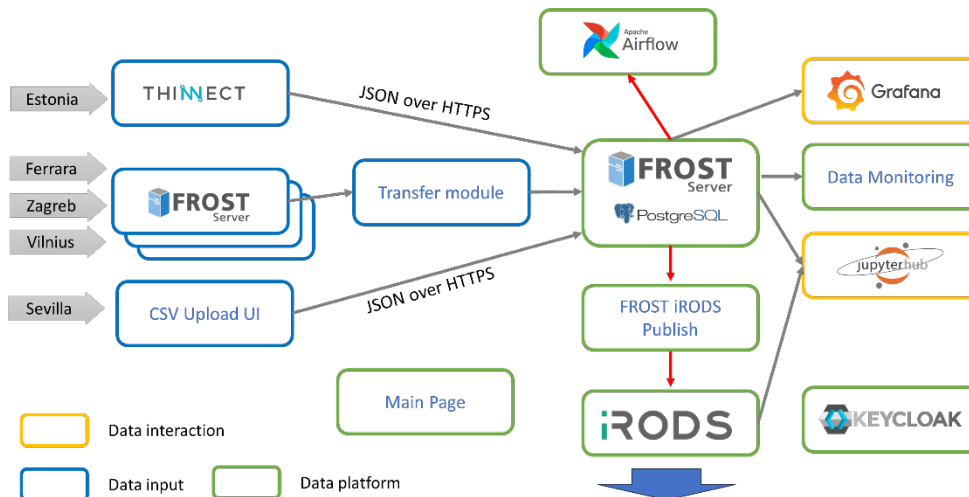


Figure 3: The components of the KDP and the dataflow between them.

Each pilot contributes data through different methods:

- **Estonia** relies on Thinnect to send data in JSON format over https to the Know-Center FROST-Server.
- **Ferrara, Zagreb and Vilnius** employ their own respective FROST Servers which stores sensor data locally. This data is then transferred and synchronized to the central Know-Center FROST-Server using a Transfer Module.
- **Sevilla** uses a custom developed CSV Upload GUI, allowing .csv files with sensor data to be manually uploaded. The .csv files are also transformed to SensorThings API standard in JSON format and are being sent over https to the central FROST-Server.

To bridge the gap between the central FROST-Server and the iRODS filesystem, the FROST iRODS Publishing is implemented as a user-friendly graphical interface. This GUI allows users to seamlessly select specific time-series data from the FROST-Server Datastreams and generate a corresponding data dump. This dump is then uploaded to iRODS, where metadata and a DOI is assigned to it. The stored data in iRODS becomes readily accessible

for further interaction, for instance if it is part of a publication or a collaboration with other researchers.

JupyterHub is the central point for data analysis as it can access data from iRODS as well as the FROST server. Researchers can use JupyterHub to conduct in-depth analytics on the data, utilizing its interactive computing capabilities for collaborative exploration of the stored sensor data.

All other components access the data in the FROST server, namely, the data monitoring, the data quality check and the visualisations.

2.2. Base Platform Architecture

The design of the platform allows the handling of two different functionalities:

- Data Discovery & Management - Handles the organization and storage of data
- Data Analytics – Focuses on extracting insights and trends from data

Data Discovery & Management

The heart of the KDP is the iRODS data management system that provides a unified interface for accessing, managing, and sharing data. iRODS consists of two different server components. One of them is the catalog provider which is responsible for managing the data catalog of an iRODS zone. The data catalog contains information about the file names, locations and permissions. An iRODS zone is a logical partition within an iRODS installation and each zone has its own settings, security rules and user permissions. The other server component in an iRODS are easily expandable data servers called catalog consumer. The catalog consumers retrieve metadata about data objects and collections from the catalog provider. Despite consisting of multiple servers, the platform appears as a single virtual file system to users. Moreover, the platform allows strong support for metadata, data-centric rules, and policies to automatically perform tasks and enforce defined policies. iRODS is



extensible to future storage technologies like Amazon S3⁸, ensuring that the platform remains up to date with advances in data storage and management.

To provide a user-friendly interface for data management tasks, the platform uses Metalnx as the graphical user interface. Users can upload, download, and share data, as well as set permissions and manage metadata. Metalnx is highly customizable with the ability to add custom metadata fields and workflows to fit the specific needs of the platform.

The Frost-Server is another central component of the KDP, enabling the efficient management and retrieval of time-series sensor data in compliance with the OGC SensorThings API. It stores sensor observations in a PostgreSQL database, ensuring high performance and scalability for spatiotemporal queries.

The authentication system is powered by Keycloak⁹ to ensure secure access to the KDP. Keycloak provides centralized authentication and authorization management, including single sign-on, multi-factor authentication and role-based access control. It is common for each data discovery and management component to utilize a PostgreSQL¹⁰ database for storing information such as user accounts, metadata on data collections and configs.

Data Analytics

The analytics part of the platform exposes JupyterHub to support collaboration and data analysis. JupyterHub is a web-based platform with a shared environment for running Jupyter notebooks. The platform allows partners and researchers to analyze and work with shared data, promoting collaboration and efficiency. Results and findings can then be easily shared among team members and beyond.

The KDP is designed to be a powerful and user-friendly data management and analytics platform that utilizes many different components. Its robust features provide researchers

⁸ <https://aws.amazon.com/de/s3/>

⁹ <https://www.keycloak.org/>

¹⁰ <https://www.postgresql.org/>



with an efficient and secure way to manage and analyse their data while also facilitating collaboration and discovery.

2.3. Data Integration

The KDP supports several methods to access and share data. This can be file-based data in iRODS or sensor measurements in the FROST server. These methods are further explained in Chapter 4 of this document.

Transfer data to KDP

Metalnx is the user interface of iRODS and provides a way for users to interact with the platform and perform common data management tasks such as uploading and downloading files, creating and managing collections, setting permissions, and searching for data.

Metalnx is connected to iRODS and built with the robust Jargon Extensions API¹¹. Using this graphical user interface is a great option for novice users, as it is easy to use and does not require any installation or setup.

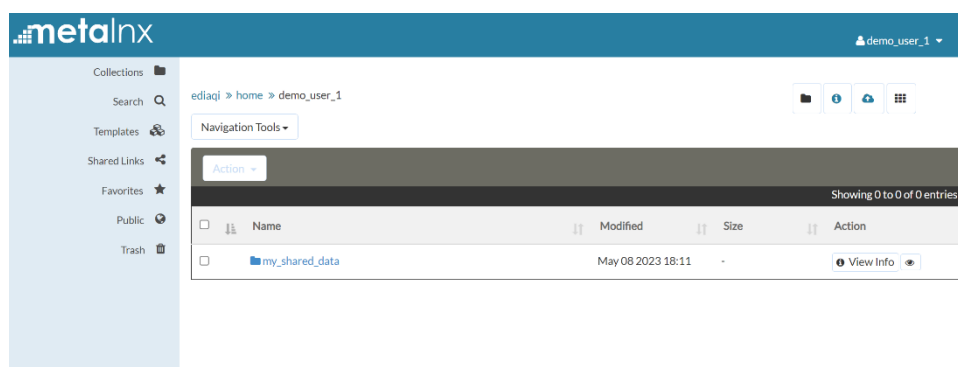


Figure 4 Landing page Metalnx GUI for the platform

¹¹ <https://github.com/DICE-UNC/jargon-irods-ext>

For users with advanced knowledge and command-line expertise, they can leverage one of the following methods to customize the data transfer to their needs.

CSV Upload GUI is a web-based application built on the streamlit framework. Users can manually upload .csv generated dynamic sensor data which will be forwarded to the Know Center FROST-Server.

FROST iRODS Publishing Bridge is a web-based GUI for intuitively selecting and filtering relevant Datastreams and uploading their corresponding Observations as a .csv file to the iRODS data management storage.

iCommands is a command-line tool to interact with the data platform. The advantage of using iCommands is that it is efficient, flexible and customizable, although it requires technical knowledge and command-line skills to use.

Setup own iRODS zone

EDIAQI partners that prefer to keep the data on their server have the option to setup an own iRODS zone and utilize secure federation with the KDP. The iRODS provider of the KDP then accepts the partner server as an additional iRODS consumer. This option enables secure sharing and accessing of data across multiple iRODS zones as if they were part of a single virtual file system. Authentication and authorization protocols ensure that data is only accessible to authorized users within their respective zones, maintaining control over access and security.



3. FAIR principles and purpose of the KNOW-Center Data Platform

The purpose of the KDP is to ensure the availability and utility of the project's research data. The KDP follows the data management plan which outlines the measures that will be taken to maximize access and re-use of the data for further purposes and applications.

In this section, we describe how the KDP supports the FAIR data principles for improving the findability, accessibility, interoperability, and reuse of research data. EDIAQI aims at creating FAIR data on both indoor and outdoor air pollutants for relevant stakeholders, so that society and public authorities have science-based, user-friendly solutions to monitor indoor air quality (IAQ), assess health risks associated to IAQ, and to prepare revised respective standards for the regulatory measures. The KDP achieves the FAIR principles by implementing several features and functionalities that support each of the four principles.

3.1. Findability

EDIAQI data will be described with rich metadata templates that link from each publication to bundles of research data. Zenodo plays a pivotal role in enhancing the findability of (meta)data. The KDP utilizes this data repository's API to assign a Digital Object Identifiers to each exported bundle. Additionally, Zenodo indexes records in its search engine to make them public findable. Uploaded datasets get versioned with Dates on the KDP. The platform utilizes Attribute-Value-Unit search to ease the discovery and search for keywords in metadata. Users can easily locate data using search queries and filters.

3.2. Accessibility

EDIAQI data will also be comprised in the four EDIAQI pilots and four EDIAQI campaigns, collated within the platform and shared on the IPCHEM platform. The KDP serves as a central access point in this data flow process. Regarding research data resource bundles of all publications will be in the "public" data section of the platform. In case of confidential information or commercial interests, partners can opt-out of public sharing of such research data. The bundles will mostly use open, text-based or binary file formats (e.g., CSV, JSON, Parquet, HDF5) for measurement data which are going to be accessible for everyone



This project has received funding from the European Union's Horizon Europe Framework Programme under grant agreement N° 101057497.

through the share feature of the KDP. Moreover, the usage of safe and open protocols like HTTPS will further contribute to the accessibility of research data.

3.3. Data Interoperability

EDIAQI data is shared through open file formats and source code in common programming languages. The in-consortium machine learning and AI-resources for data analysis will be container-based with instances of the Jupyter stack. Research data and related meta data (e.g., configurations and context) is shared through open file formats, source code in common programming languages, and GitHub repositories which allows using the default workflow most researchers in computer science adopted. EDIAQI will leverage on open standards defined by International Organization for Standardization (ISO), Comité Européen de Normalisation (CEN), Open Geospatial Consortium (OGC) and other organizations, exploiting decades of experience on data and service interoperability related to IoT and other domains. The main OGC standards considered in EDIAQI are SensorThings API, CityGML15 (or INSPIRE Buildings) and web map services. Moreover, the platform recognizes specific naming conventions and assigns appropriate filled-out metadata templates to it. Cross-references in metadata are visible in the platform's graphical user interface.

3.4. Reusability

Any shared EDIAQI data assets will be licensed under the creative commons attribution license (CC BY 4.0, <https://creativecommons.org/licenses/by/4.0/>), i.e., Creative Commons Public Domain Dedication CC0 1.0 Universal (CC0), the Creative Commons Attribution 4.0 International (CC BY 4.0), or Creative Commons Public Domain Dedication CC0 1.0 Universal license will be used. The partners are free to pick a license. However, we suggest a liberal license such as the Apache License Version 2, which allows commercial extensibility. The KDP supports the attachment of licences as metadata.



4. Usage of the KNOW-Center Data Platform

This section shows some common tasks and naming convention when using the base data platform. Each task can either be performed over the Metalnx GUI or via the command line interface using iCommands.

The Metalnx tutorials are illustrated with screenshots. The iCommands tasks are provided with a link to the documentation of the command.

4.1. Access platform & Login

Metalnx

KNOW provides each individual partner accounts on its data platform to simplify collaboration within organisations and to make onboarding as easy as possible.

Access in browser with link <https://kdp-ediaqi.know-center.at/metalnx/login/> and use credentials to login.

sign in to Metalnx

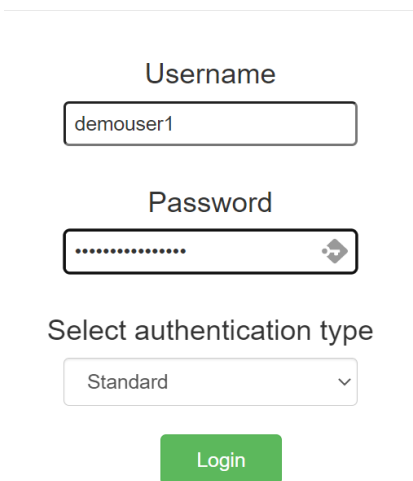


Figure 3 Login mask of metalnx

iCommands

Install iCommands on Linux or Linux Subsystem for Windows with the following commands

```
wget -qO - https://packages.irods.org/irods-signing-key.asc | sudo apt-key
add
echo "deb [arch=amd64] https://packages.irods.org/apt/ $(lsb_release -sc)
main" | sudo tee /etc/apt/sources.list.d/renci-irods.list
```



This project has received funding from the European Union's Horizon Europe Framework Programme under grant agreement N° 101057497.

```

sudo                                apt-get                                update
sudo  apt  -y  install  aptitude  irods-runtime  irods-icommands
sudo aptitude hold irods-runtime irods-icommands

```

Use iinit to configure the host/zone you want to connect to.

<https://docs.irods.org/master/icommands/user/#iinit>

```

Enter the host name (DNS) of the server to connect to: kcs-aide.demo.know-center.at
Enter the port number: 1247
Enter your irods user name: demouser1
Enter your irods zone: ediaqi
Those values will be added to your environment file (for use by
other iCommands) if the login succeeds.

Enter your current iRODS password:

```

Figure 4 Input configurations to access KDP using iinit

Note: to access the public domain kdp-ediaqi.know-center.at via iCommands, someone's IP-address needs to be whitelisted. For the IP-address to be whitelisted, individuals or organizations should reach out to the Data Management for AI group at Know Center. It should be considered to provide details such as the range of IP addresses or specific IP addresses associated with their server, ensuring that necessary information is shared to facilitate the whitelisting process effectively.

4.2. Upload, Download, Rename & Delete File(s)

Metalnx

Under Collections press the following icon button to upload one or multiple files at once.



Figure 5 Upload file(s) to iRODS button

Afterwards select the file(s) from your local storage and press "Upload". Folders cannot be uploaded.



To download file(s) and folders tick the checkbox left to the data object and go to Action > Download.

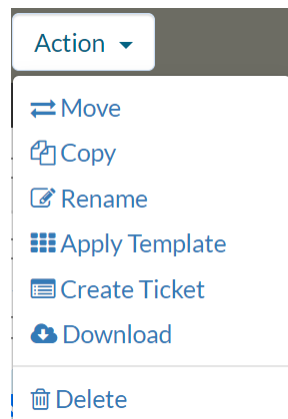


Figure 6 Action button options

The download starts immediately without any prompts. Figure 6 also shows the options to either rename or delete a file.

iCommands

Change collection:

<https://docs.irods.org/master/icommands/user/#icd>

Upload file:

<https://docs.irods.org/master/icommands/user/#iput>

Download file:

<https://docs.irods.org/master/icommands/user/#iget>

Rename data object:

<https://docs.irods.org/master/icommands/user/#imv>

Remove data object:

<https://docs.irods.org/master/icommands/user/#irm>



4.3. Add Folder, Move & Copy File(s)

Metalnx

Files can be organized into folders which are called "Collections" within iRODS terminology. A path is a specific location or address that helps identify the location of a file or folder within a file system. In the context of iRODS, a path specifies the hierarchical structure of collections and sub-collections. It represents the sequence of folders or directories you need to traverse to reach a specific file or folder. For example, the creation of a collection named "new-folder" within the "user" folder which is in the "ediaqi/home" directory, will produce a path like "/ediaqi/home/user/new-folder". To create a collection, navigate to the location where the folder should be created and click on the corresponding icon as shown in the figure below.



Figure 7 Add Collection button

A prompt will open for the name of the collection. Then press "Save" to create the folder. Collections and files can also be copied or moved to a different path. Mark the wished file(s) & collection(s) to be moved and press the Action > Move or Copy option (visible also in Figure 6). Afterwards a dialog will open, asking for the target location.

Note: It can also be moved to the "Public" section where everybody has access to it.

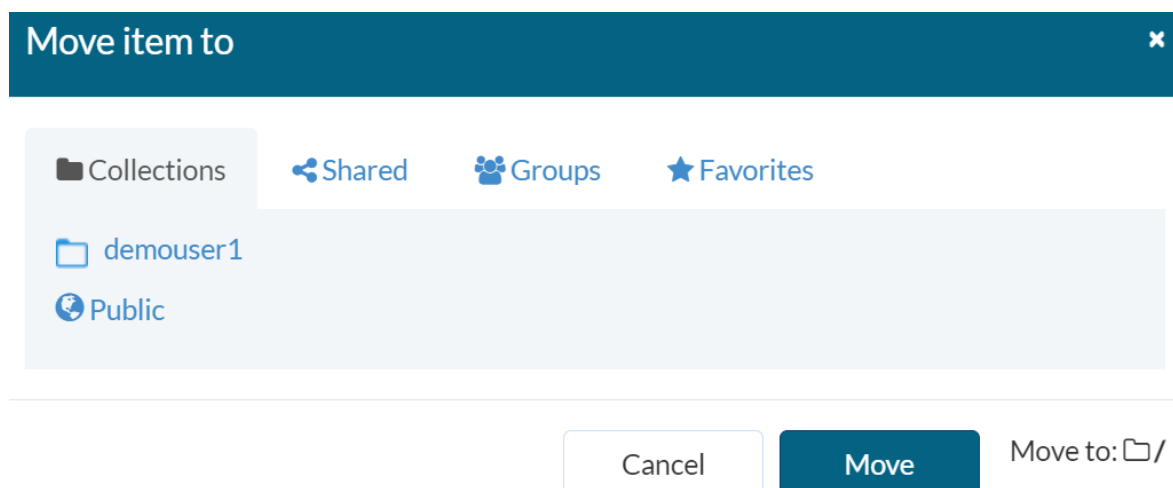


Figure 8 Choose where the file(s) should be moved to

iCommands

Create a collection:

<https://docs.irods.org/master/icommands/user/#imkdir>

Move data object:

<https://docs.irods.org/master/icommands/user/#imv>

Copy data object:

<https://docs.irods.org/master/icommands/user/#icp>

4.4. Create Templates & Add Metadata

Metalnx

In the sidebar the option Templates offers to manage metadata templates which then can be filled out when applying it to a data object. Templates serve as predefined structures that assist users in creating consistent and organized data structures. In the context of Attribute-Value-Unit (AVU) structures, templates define specific attributes, their corresponding values and associated units. This ensures that data is captured uniformly and can be easily processed. For instance, consider a template for environmental sensors. This template could be structured in the AVU format as follows:

```
Attribute: Sensor ID; Value: S12345
Attribute: Location; Value: Room A
```



Attribute: Measurement Type; Value: Temperature; Unit: Celsius

By associating metadata with the sensor, users can better understand its specifications, capabilities and context.

To create a new one, click the Templates > “Add Template” button.

Template Settings

*Template Name

Sensor

*Description

Describes the sensor from which the data arised.

*Access

Private

Only demouser1 will view/edit/apply this template.

Metadata

+ Metadata

☐ Attribute	Value	Unit
☐ Location		
☐ Sampling Rate		
☐ Sensor ID		
☐ Sensor Type		

Figure 9 Example template for a sensor

This template can then be applied to file(s)/collection(s) by ticking them and choosing Action > Apply Template.

Step 2 - Fill-in the values

Showing 1 to 4 of 4 entries

Attribute	Value (requir...	Unit	Template
Location	Zagreb	city	Sensor
Sampling Rate	25	seconds	Sensor
Sensor ID	1	numeric	Sensor
Sensor Type	CO2		Sensor

Back
Cancel
Apply Template

Figure 10 Fill-in the values for chosen template

The data object its metadata can then be further fine grained by clicking on the View Info on the right of the file and then selecting the Metadata tab. The “+Metadata” button can be used to apply even more AVU metadata to the data object.

iCommands

Add metadata when uploading via --metadata flag:

<https://docs.irods.org/master/icommands/user/#input>

Adding/Viewing metadata of collection/data object:

<https://docs.irods.org/4.2.0/icommands/metadata/#imeta>

4.5. Permissions & Shared Links

Metalnx

Collections and their files in it can be shared with other users and groups. This feature is available under the “View Info” on the right of the collection and then under the permissions tab. The “+Permissions” button can be used to add more users and groups.



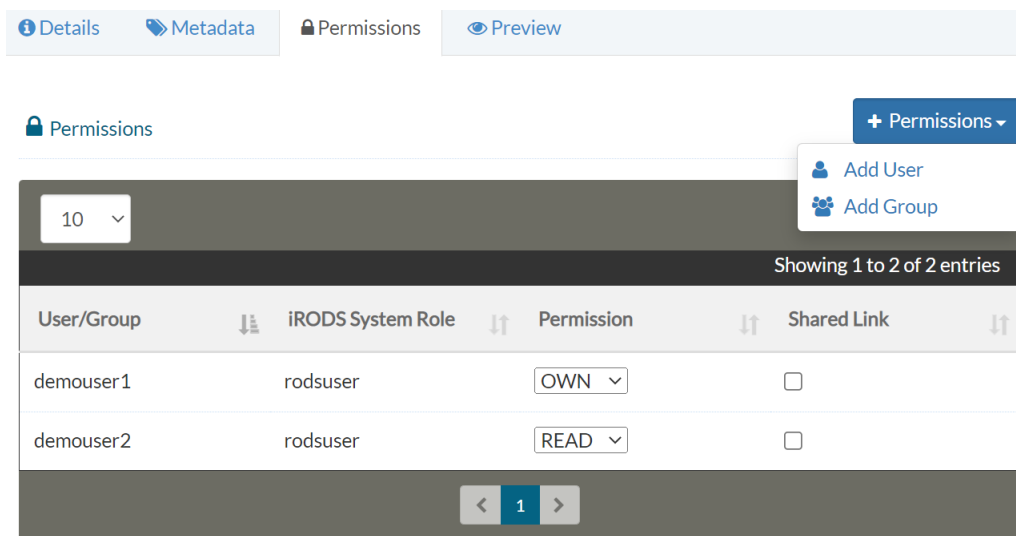


Figure 11 Manage permissions of a collection

Afterwards the shared collection is accessible in the Shared Links menu for the chosen user/group.

There is also the option to apply iRODS inheritance to a collection when either creating it or ticking the inheritance box under View Info > Details tab. When inheritance is applied to a collection, any permissions set on the parent collection are automatically inherited by all its subcollections, simplifying the management of access control.

For example, the parent collection, "shared_folder," is created and assigned specific permissions. User1 is granted Read-Write access, allowing them to view and modify the collection's contents, while User2 is granted Read access, allowing them to only view the contents. When subcollections are created under "shared_folder" such as "data_set1" and "data_set2," the inheritance option is enabled for both. This means that the permissions from the parent collection automatically propagate to the subcollections. Consequently, User1 retains Read-Write access to both "data_set1" and "data_set2," and User2 maintains Read access.

iCommands

Manage permissions of collections and set inheritance:

<https://docs.irods.org/master/icommands/user/#ichmod>



4.6. Jupyterhub

JupyterHub is a multi-user platform that allows you to create and share documents containing live code, equations, visualizations and narrative text. It is designed to provide access to Jupyter notebooks in a scalable environment, supporting multiple users, making it ideal research teams.

Supported Kernels

JupyterHub supports multiple programming languages via kernels. Two of the most used kernels in JupyterHub are:

- **Python:** A versatile programming language known for its ease of use and powerful libraries, particularly in data science, machine learning, and automation.
- **R:** A programming language and environment specifically designed for statistical computing and graphics, widely used in data analysis and research.

This flexibility allows users to leverage the strengths of both languages within the same collaborative environment.

Example Notebooks

In the EDIAQI project we provide for each account four example notebooks to help users get started with common tasks related to data retrieval in the know data ecosystem. These notebooks are split between Python and R languages, ensuring that users of both languages have relevant examples to learn from. In each language they show how to:

- **Get Data Over HTTP requests from FROST-Server:** This notebook demonstrates how to connect to and retrieve data from the KNOW-FROST-Server using HTTP requests. It provides step-by-step instructions for querying multiple datastreams and visualizing the data.



- **Get Data from iRODS System:** This notebook walks users through connecting to an iRODS data management system, retrieving the uploaded dumps from the FROST iRODS bridge and performing basic data manipulations and analysis.

These notebooks serve as practical examples for users to understand the basics of working with data from FROST-Server and iRODS, and can be easily adapted to suit specific needs or extended for more complex tasks.

Additional Resources

For more detailed information on how to configure and use JupyterHub, please refer to the official JupyterHub documentation: [JupyterHub Documentation](#)



5. Usage of Data Support Interoperability Tools

5.1. CSV Upload GUI

The CSV Upload to FROST Server web application allows users to upload CSV files containing observational data and map these data to existing datastreams within the KNOW FROST-Server. This tool is particularly useful for integrating external data sources that are stored in .csv files into the FROST-Server.

Login and Select a Thing

- After logging in, users can select the Thing entity from the dropdown list in which they want to upload their Observations.
- A brief description of the selected Thing entity and the list of associated datastreams is displayed for reference.

Upload CSV File

- Drag and drop the CSV file into the upload area or click "Browse files" to select a file manually.
- The file must be in CSV format, with a size limit of 1GB. Users can adjust the delimiter if the CSV file uses a different separator (default is a comma).

Mapping Value Columns with Datastreams

- The system attempts to automatically map the value columns in the CSV file to the corresponding datastreams.
- If it cannot find a match, users are prompted to manually map the value columns to the correct datastream using the provided dropdown menus.



Data duplicates and finalizing Upload

- Once all value columns are correctly mapped to datastreams, users can proceed with the data upload. There is a check if the data was already imported in the FROST-Server to avoid duplication.
- The system will then push the data to the FROST-Server, where it will be stored and made available for further querying and analysis.

Additional Options

- **View Schema:** Users can view the schema of the SensorThings API to get a better understanding of the structure in the FROST-Server.
- **View Uploaded Observations:** Allows users to review previously uploaded data, ensuring that they have visibility into what data has already been integrated into the FROST-Server.
- **View Missing Data:** Helps users identify gaps in the data, ensuring that all necessary observations are uploaded.

5.2. Frost iRODS Publish Bridge

The FROST iRODS Bridge web application provides users with the ability to hierarchically query and select multiple data streams Observations from the KNOW-FROST-Server and subsequently upload the selected data to the iRODS data management system. This tool assigns a DOI to each specified dump that is successfully uploaded to the platform.

Filter by Thing, Sensor, and Datastream

- **Thing(s):** Choose the specific 'Thing' (e.g., a particular device or environmental entity) from the available options related to the selected pilot.
- **Sensor(s):** Select the appropriate sensor associated with the chosen thing.



- **Datastream(s):** Finally, select the datastream that corresponds to the chosen sensor. The datastream represents the actual data being generated, such as temperature readings, humidity levels, or particulate matter (like PM10) concentrations.

Select a Date Range

Use the date picker to specify the timeframe for the data you wish to retrieve. The system allows you to set a start and end date, ensuring you capture the exact period of interest on which the phenomenon Time is being filtered on.

Preview the Data

Optionally, check the "Show Preview" box to view a preview of the selected data in CSV format. This step helps in verifying the data before uploading it to the iRODS system.

Publish to iRODS

- After confirming the selection, provide a name for the CSV file in the "Know Data platform home user path" field. This name will be used to save the file within the iRODS system. Folders inside the /home/ are also valid.
- Click the "Publish .csv file to Know Data Platform" button to upload the selected data. The system will create the .csv file and store it at the specified path within the iRODS platform.

Following screenshot captures these explained options.



EDIAQI Obtain SensorThings Observations from Know-Center FROST-Server

Select datastream by filtering

Select Pilot or Campaign
Ferrara Pilot

Select Thing(s)
THI.FE.001 x

Select Sensor(s)
SEN.001 x

Select Datastream(s)
PM10@THI.FE.001 x

Select a date
2023-08-01 2024-05-02
2023-08-01 2024-05-02

☐ Show Preview

Know Data platform home user path
PM10_THI_FE_001.csv

Publish .csv file to Know Data Platform

Figure 5 Select datastreams and upload their Observations as dump to iRODS

5.3. Data Monitoring

The EDIAQI Data Monitoring dashboard provides a comprehensive overview of the ongoing projects, the data being collected, and the various sensors deployed across different locations. This interface is designed to give users a quick and clear understanding of the status and distribution of data streams, things, locations and sensors involved in each project. The monitoring capabilities are important for managing large-scale environmental data collection initiatives and ensuring data quality.

Overall Monitoring

- Shows the total number of Things, Locations and Sensors.



- Shows a pie chart distribution for Things and Locations.
- Displays datastreams count in each project
- Display datastreams in each Project and Sensor

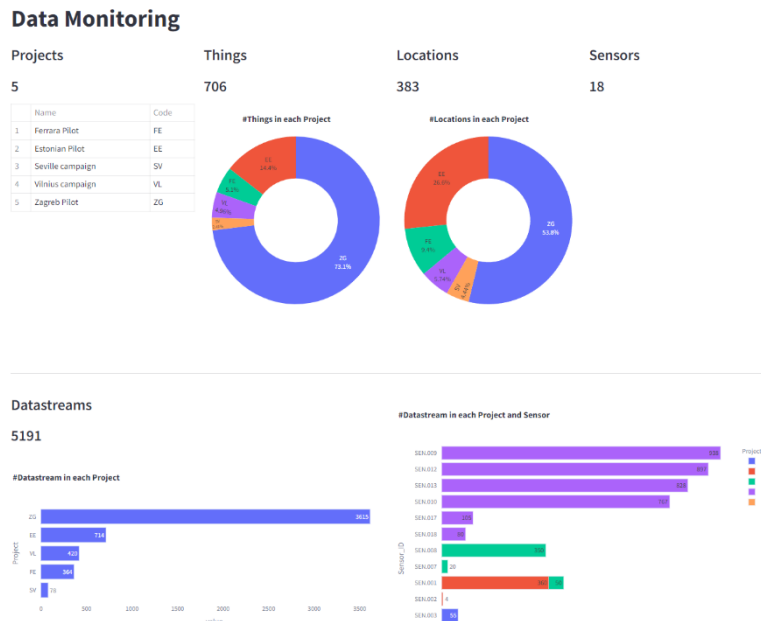


Figure 6 Data monitoring statistics in overall EDIAQI

Pilot Monitoring

Displays a heatmap for each datastream to see if there are any gaps where no Observation data has been provided. Additionally, a list of all Things with no Observations is returned. Also a list of all single Datastreams that are not streaming is displayed.

Data Monitoring ^{GO}

Things

Locations

36

36

Observations per Day

#Datastream has Observations: 327



Figure 7 Identify gaps in each pilot datastreams

Check Duplicates

Displays datastreams that contain deduplicated Observations.

Duplicates detected

NAME	#Total	#Deduplicated	Diff
CO@TH.FE.015	50,803	36,404	14,399
O@TH.FE.015	50,803	36,404	14,399
NO2@TH.FE.015	50,803	36,404	14,399
O@TH.FE.001	2	1	1

#Observation in Database

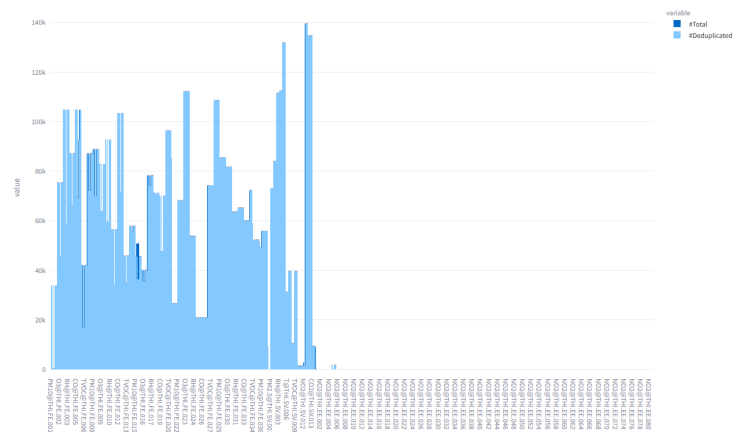


Figure 8 Amount Observations in each datastream



View Data in Details

Select a Datastream to visualize and inspect

- The Observation Count per day or in a period
- The value plot over a specific period



Figure 9 TimeSeries for Temperature datastream in period from November to January

Check Begin and End time

In this section, for each Thing, the static data **beginDatetime** and **endDatetime** are parsed and compared with the first and last Observation of any Datastream linked to the Thing. The goal is to check whether the static data is (1) available, (2) if it can be parsed and (3) if the observation do approximately match with the planned schedule.

5.4. Data Troubleshooting

For all datastreams in the FROST-Server, these features are calculated and displayed in heatmaps:

- Percentage of zeros
- Percentage of negative values
- Number of unique values (to check quantization)
- Number of observations



- Streaming percentage
- Sampling interval

When visualizing the heatmap it is possible to detect outliers and to gain a good overview of problems and data quality across all the data of the project. Furthermore, the mean value, the CO2-baseline and other statistics are computed for all datastreams and displayed in bar plots.

negativeValues[%]

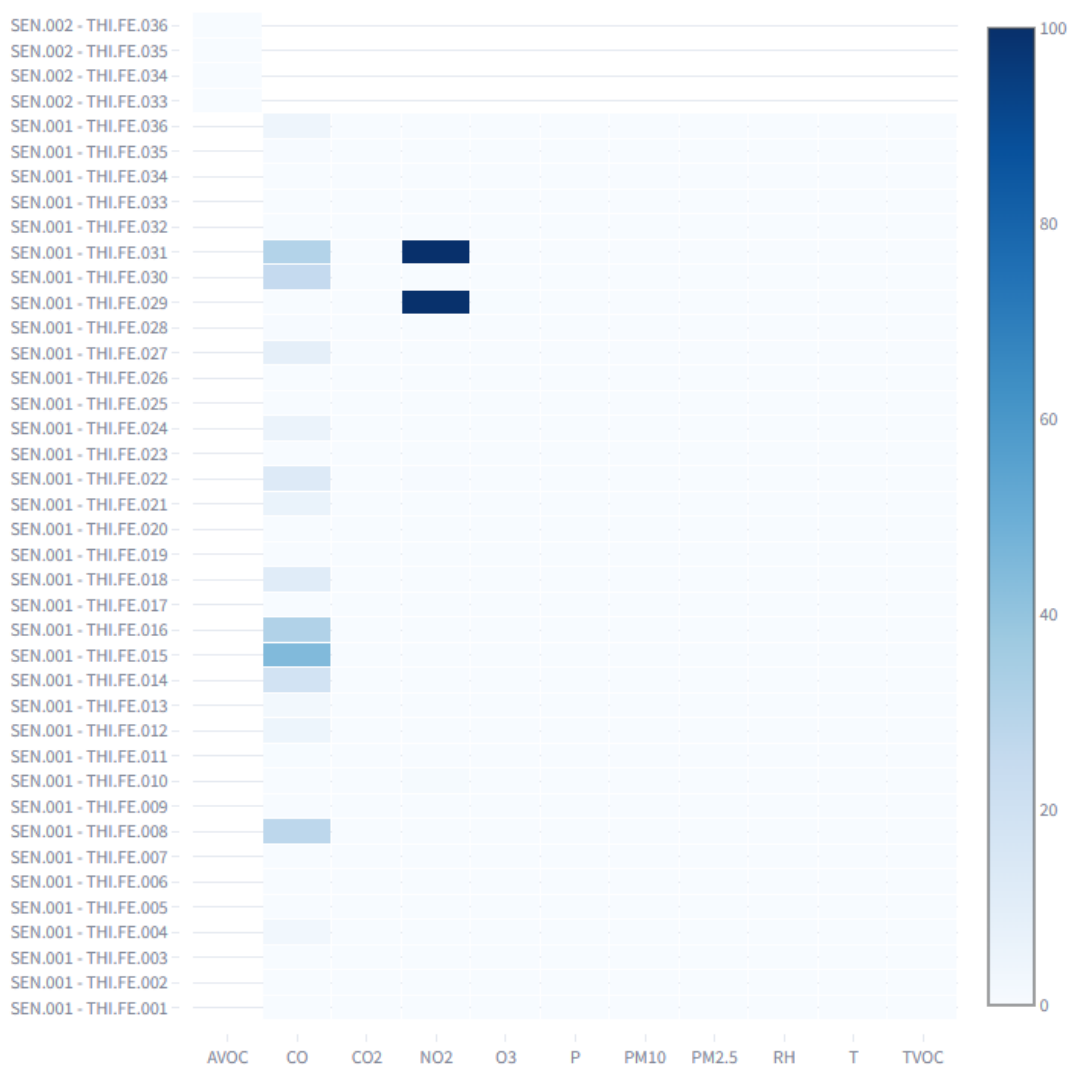


Figure 10 Datastreams with negative percentage values.

AQ Data Comparison

For given thresholds, the Air Quality of the datastreams and things is calculated and displayed in pie plots and percentage bar plots.

For similar things (e.g. classrooms) the AQ can be compared across the various pilots.

It is possible to set conditions (e.g. omit school holidays and/or weekends) on the dataset and to partition it for visualization (e.g. partition over months).

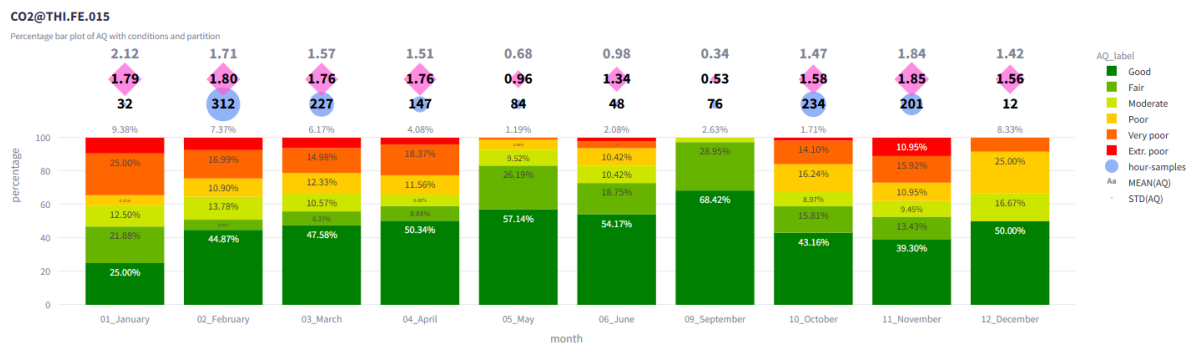


Figure 11 Percentage bar plots for CO2 in given datastream.



Deliverable D4.8

GitHub Repository of EDIAQI Models Code - Version 2

Work Package 4

MONITOR – Pilots, data and campaigns

Version: Final



This project has received funding from the European Union's
Horizon Europe Framework Programme under grant agreement N° 101057497.