K-HEALTH inAIR

K-HEALTHinAIR

Knowledge for improving indoor AIR quality and HEALTH

D4.1 K-HEALTHinAIR BigData platform (Version I)

Funded by the European Union

Project Title	Knowledge for improving indoor AIR quality and HEALTH
Project No	101057693
Contract start date	01/09/2022
Contract duration	48 Months

Document ID	K-HEALTHinAIR _D4.1 K-HEALTHinAIR BigData platform (Version I) V1.0
Deliverable leader	ATOS
Due date	31/08/2023
Deliverable date	17/08/2023
Dissemination level	PUBLIC

AUTHORS – CONTRIBUTORS

Name	Organization
Roberto Hernández Ruiz	ATOS

PEER – REVIEWERS

Name	Organization
María Figols	INBIOT
Suzanne van den Toren / Simon de Leede	ERASMUS MEDICAL CENTER
Mireia Ferri	KVELOCE I+D+i

Funded by the European Union

DOCUMENT HISTORY

Version	Date	Author/Contributor (Organization)	Modifications	Status
V0.1	01/06/2023	Roberto Hernández Ruiz (ATOS)	Initial redaction	Draft
V0.2	12/07/2023	Roberto Hernández Ruiz, Alberto Acebes, César Mediavilla (ATOS)	Improvements and updates	Draft
V0.2_INB	20/07/2023	Maria Figols (INB)	Chapter 3 revision	Draft
V0.3	26/07/2023	Roberto Hernández Ruiz (ATOS)	Final version for reviewers	Draft
V0.3_KVC	31/07/2023	Mireia Ferri Sanz (KVC)	WP leader review	Revision
V0.3-SvdT_SdL	02/08/2023	Suzanne van den Toren / Simon de Leede (EMC)	Internal review	Revision
V0.3-SvdT_SdL-INB	15/08/2023	María Figols (INB)	Internal review	Revision
V1.0	17/08/2023	Roberto Hernández Ruiz (ATOS)	Final version for submission	Submitted

Disclaimer

This deliverable may be subject to final acceptance by the European Commission. The information and views set out in this document are those of the authors and do not necessarily reflect the official opinion of the European Commission. Neither the Commission nor any person acting on the Commission's behalf may hold responsible for the use which may be made of the information contained therein.

Copyright message

Funded by the European Union

Funded by the European Union

TABLE OF CONTENT

Funded by the European Union

LIST OF FIGURES

LIST OF TABLES

Funded by the European Union

ABBREVIATIONS AND ACRONYMS

ACRONYMS	DESCRIPTION
API	Application Programming Interface
AQ, IAQ, OAQ	Air Quality, Indoor AQ, Outdoor AQ
AQDM	Air Quality Data Model
AQDR	Air Quality Data Repository
CDR	Clinical Data Repository
DAQSR	Discrete Air Quality Sampling Repository
DB, TSDB	Database, Time-Series Database
DOA	Description Of Action
ETL	Extract, Transform, Load
HDH	Health Data Hub
ID	Identification
INB	InBiot Monitoring SL
K8S	Kubernetes
M+H	Mann+Hummel GmbH
SDK	Software Development Kit
UI	User Interface

Funded by the European Union

EXECUTIVE SUMMARY

This deliverable D4.1: K-HEALTHinAIR BigData Platform (Version I) reports the current state of the Open Access Platform components of the K-HEALTHinAIR project at month 12. It presents the results and work related to the Task 4.1: Platform specification and construction (M1-M48). Thus, this is only the first version of the K-HEALTHinAIR Platform design, since additional versions of this deliverable are scheduled throughout the project (Version II at month 24, Version III at month 36, Version IV at month 48).

The K-HEALTHinAIR Platform is built following a microservices-based architecture and strongly depends on the current state and on the input provided from the *WP1: IAQ systematic analysis and monitoring for 9 relevant scenarios (5 pilot studies)*. The platform includes several related modules that help to collect, transform and harmonise all the data that is considered relevant for the project, as well as to enable its visualisation and persistence during the entire project lifecycle. The K-HEALTHinAIR Platform components (an Air Quality Data Repository for continuously monitored indoor and outdoor air quality measurements, a Discrete Air Quality Sampling Repository for discrete measurements that are periodically retrieved, and a Health Data Hub that hosts all clinical or health data from participants in the project) are described in this document. The entire design has been approached considering future data exploitation and analysis.

Funded by the European Union

1. Introduction

This document reports on the architecture and specifications of the K-HEALTHinAIR Open Access Platform and the technical work carried out during the first 12 months of the project. As the K-HEALTHinAIR Description of the Action (DoA) states, this open access platform will include data collection, transformation, harmonisation, and persistence functionalities.

After this introduction (Chapter 1), Chapter 2: K-HEALTHinAIR Platform overview shows the different parts that integrate the entire K-HEALTHinAIR Platform, explaining their specific purposes, features and functionalities, and giving a high-level overview of their maturity level.

The designed Identification (ID) Convention for the different pilots, scenarios, discrete measurements and participants in the project is explained in Chapter 3: K-HEALTHinAIR Platform conventions. The Air Quality Data Model (AQDM) is presented in this section too.

Chapter 4: K-HEALTHinAIR Platform components provides the links of the working modules of the platform. A user guide for the Air Quality Data Repository (AQDR) is also illustrated in Chapter 5: K-HEALTHinAIR AQDR User Guide. Finally, some conclusions of this work and next expected steps for the coming months are exposed in Chapter 6: Conclusions.

Funded by the European Union

2. K-HEALTHinAIR Platform overview

The implementation of the different components of the K-HEALTHinAIR Open Access Platform follows a microservices-based architecture[1] paradigm. This approach is perfectly aligned with the most innovative and current software development practices and architecture designs. A microservices-oriented architecture enables a complete independence between different smaller components within a large application, each one of them being responsible for their own tasks and purposes. This also facilitates the final environment configuration without depending on a single service, as well as the deployment of the platform over any type of infrastructure. Thus, all the modules that form K-HEALTHinAIR Open Access Platform will be containerised and isolated using Docker[2] containers and Kubernetes[3] (k8s) as container orchestration platform.

The K-HEALTHinAIR Platform will include the following components that are illustrated in *Figure 1*.

- Air Quality Data Repository (AQDR). This component manages both Indoor (IAQ) and Outdoor (OAQ) Air Quality data from the continuous monitoring devices of InBiot (INB) and Mann+Hummel (M+H). It consists of the AQDR Time-Series Database (TSDB), the AQDR Ingestor, and an AQDR Application Programming Interface (API) to facilitate data retrieval from the AQDR for data analysis purposes.

- Discrete Air Quality Sampling Repository (DAQSR). It stores all the data from the discrete samples that will be periodically carried out during the project lifecycle: Volatile Organic Compounds (VOC) samples, Particles Matter (PMs) samples, microbiome samples, formaldehydes, etc.

- Health Data Hub (HDH). This third component manages clinical and health data of the participants and centralises the health information. It is, in turn, divided into a Clinical Data Repository (CDR) and a Clinical Data Repository API (CDR API).

[1] Microservices architecture, a variant of the service-oriented architecture style (*https://en.wikipedia.org/wiki/Microservices*)
[2] Docker, an open platform for developing, shipping, and running applications (*https://www.docker.com*)
[3] Kubernetes, an open-source system for automating management of containerised applications (*https://kubernetes.io*)

Funded by the European Union

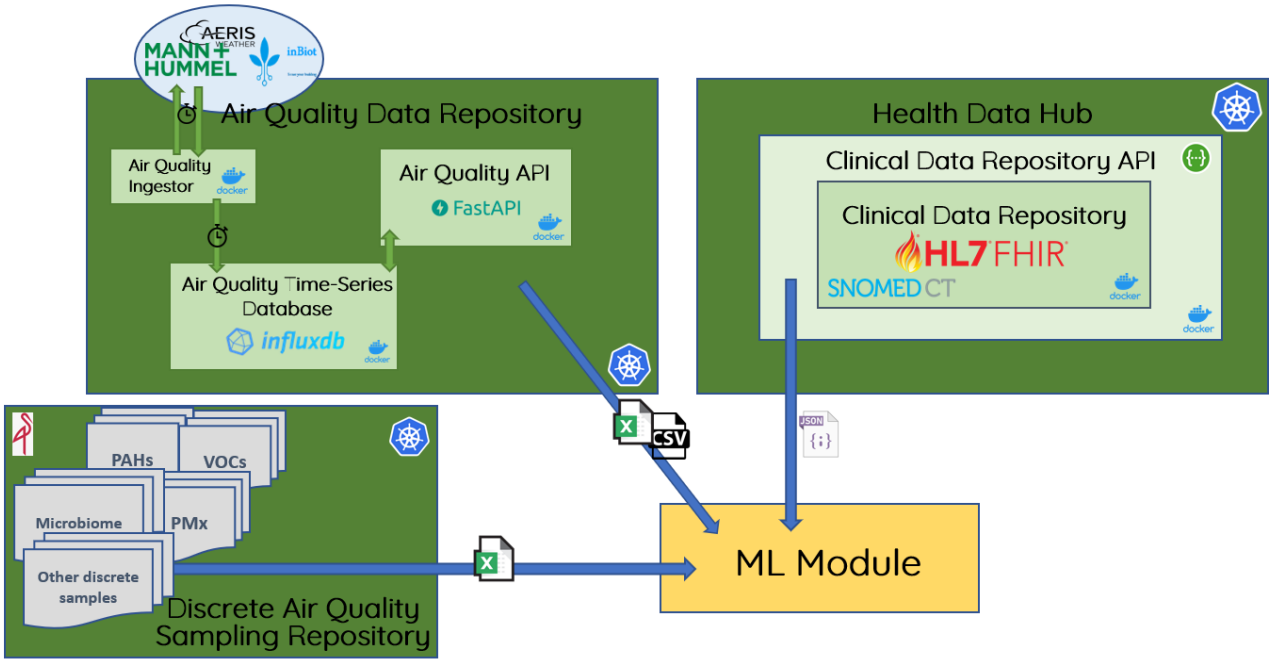

Figure 1: K-HEALTHinAIR Open Access Platform main architecture.

2.1. Air Quality Data Repository (AQDR)

The Air Quality Data Repository (AQDR) manages the air quality information that comes from the continuous monitoring devices provided by INB and M+H. It includes three modules (see *Figure 2*), and it is based on K-HEALTHinAIR Air Quality Data Model (AQDM):

- AQDR Database, which is based on InfluxDB[4], a Time-Series Database. This module stores all IAQ and OAQ records continuously monitored by INB and M+H sensors, thanks to the ETL (Extract, Transform, Load) processes that AQDR Ingestor performs.

- AQDR Ingestor, which is the module responsible for running all scheduled ETL pipelines. The name is derived from its function, as it performs the automated ingestion of the data collected by the sensors into the K-HEALTHinAIR project repository. This automated ingestion is managed through three different ETL processes that are described later.

- AQDR API, which is based on FastAPI[5]. This module enables direct communication with AQDR Database and facilitates some GET[6] endpoints to retrieve AQ data that is stored in it.

[4] InfluxDB, an open-source time series database (*https://www.influxdata.com*)
[5] FastAPI, a modern, fast high-performance, web framework for building APIs with Python (*https://fastapi.tiangolo.com*)
[6] HTTP GET, the HTTP method for requesting data from a server.

Funded by the European Union

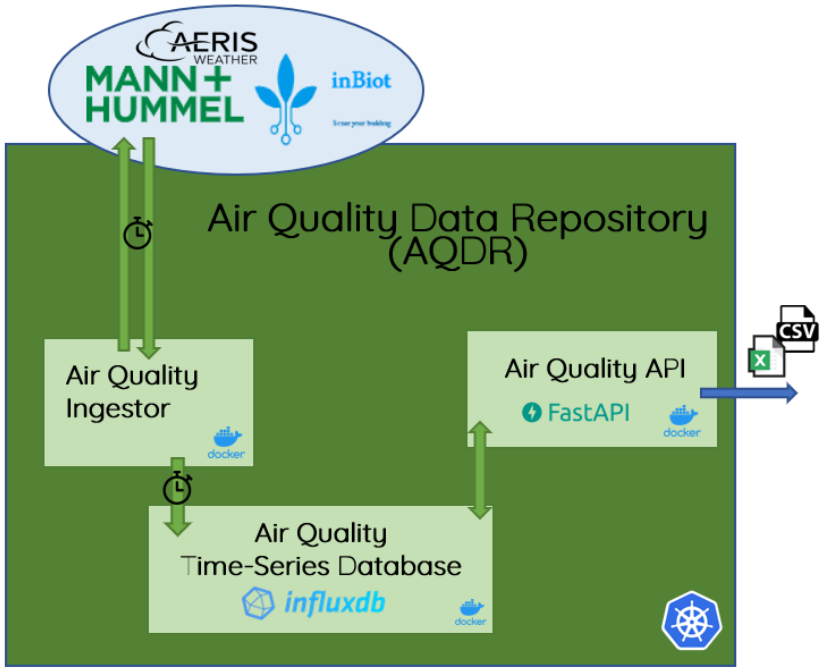

Figure 2: AQDR basic architecture (AQDR DB, AQDR Ingestor, AQDR API).

2.1.1. AQDR Database

As mentioned before, AQDR Database consists of a time-series database where continuously monitored IAQ and OAQ data is stored. It is based on an InfluxDB instance. Thus, some InfluxDB-specific terminology is firstly explained here:

- *Organization*: a workspace for a group of users that share dashboards, tasks, databases, etc. K-HEALTHinAIR AQDR DB is located in 'K-HIA' organization in our InfluxDB instance of the platform.
- *Bucket*: a named location where time series data is stored, which is the first level of hierarchy of InfluxDB. A bucket belongs to an organization, and it could contain several databases with different schemas. All buckets in InfluxDB have a retention period that indicates how much time data persists, which in K-HEALTHinAIR AQDR is set to 'forever'. K-HEALTHinAIR AQDR DB resides in 'AIR_QUALITY' bucket within the previously mentioned organization in the instance.
- *Measurement*: analogous to a specific database table in a regular SQL-DB context, i.e., the TSDB in the strict sense, where datapoints will be registered. It acts as a descriptor for the data that is stored in it.

So, AQDR Database is initially positioned within an organization ('K-HIA') and bucket ('AIR_QUALITY') as a so-called measurement ('CONTINUOUS_MONITORING') in our InfluxDB-based environment. Our K-HEALTHinAIR AQDR DB displays a powerful user-friendly interface (UI) that shows the evolution of the registered measurements over time (see *Figure 3*). Further

Funded by the European Union

functioning from the user's perspective is explained in Chapter 5: K-HEALTHinAIR AQDR User Guide.

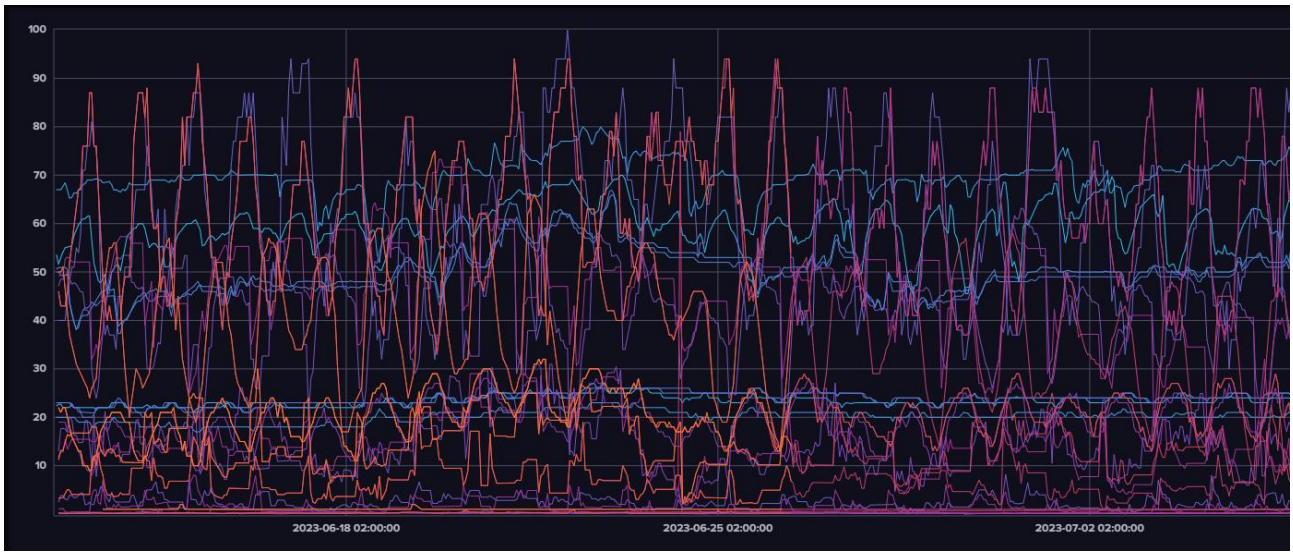

Figure 3: AQDR Time-Series Database for indoor and outdoor air quality data.

2.1.2. AQDR Ingestor

Secondly, AQDR Ingestor is the module within AQDR that is responsible for automating the scheduled batches or iterations of the 3 different ETL processes that are here explained:

1. Outdoor AQ ETL Pipeline, executed every 60 minutes.
2. InBiot Indoor AQ ETL Pipeline, executed every 10 minutes.
3. Mann+Hummel Indoor AQ ETL Pipeline, executed every 5 minutes.

These periodicities depend on each sensor provider and are initially configured as specified above. However, the periodicities of each pipeline can be modified independently during the course of the project. In the case of OAQ, this information is retrieved through M+H API from AerisWeather[7] platform.

All three ETL processes follow the same steps:

- Extract: AQDR Ingestor retrieves last available record for each sensor through the correspondent provider's API, following given guidelines from those APIs.

- Transform. In this step, a raw response from the previous GET request (Extract step) is received. The raw data is then transformed according to K-HEALTHinAIR AQDM and several pre-established schemas, to convert it to a suitable format for the AQDR DB (i.e., to convert it to an InfluxDB datapoint).

- Load. In this final step, the already processed datapoint is ingested in the TSDB.

[7] AerisWeather, the Global Weather API & Mapping Platform for Business (*https://www.aerisweather.com*)

Funded by the European Union

In all these ETL processes, only K-HEALTHinAIR valid sensors are considered. To this end, each sensor device first passes some validation tests that ensure that the provided information is K-HEALTHinAIR ID conventions- compliant (see Sensors Description File).

- AQDR Back-up mechanisms

Furthermore, AQDR Ingestor includes a back-up mechanism on stand-up. These processes are launched every time the AQDR (i.e., AQDR DB module together with the Ingestor module) instance is either restarted or initialised from scratch. Its main goal is to retrieve past measurements that could have been lost (and therefore could have not been ingested in AQDR) while the app was unexpectedly not working, as an automated data recovery. The proposed strategy for this back-up strategy follows several steps (see *Figure 4*):

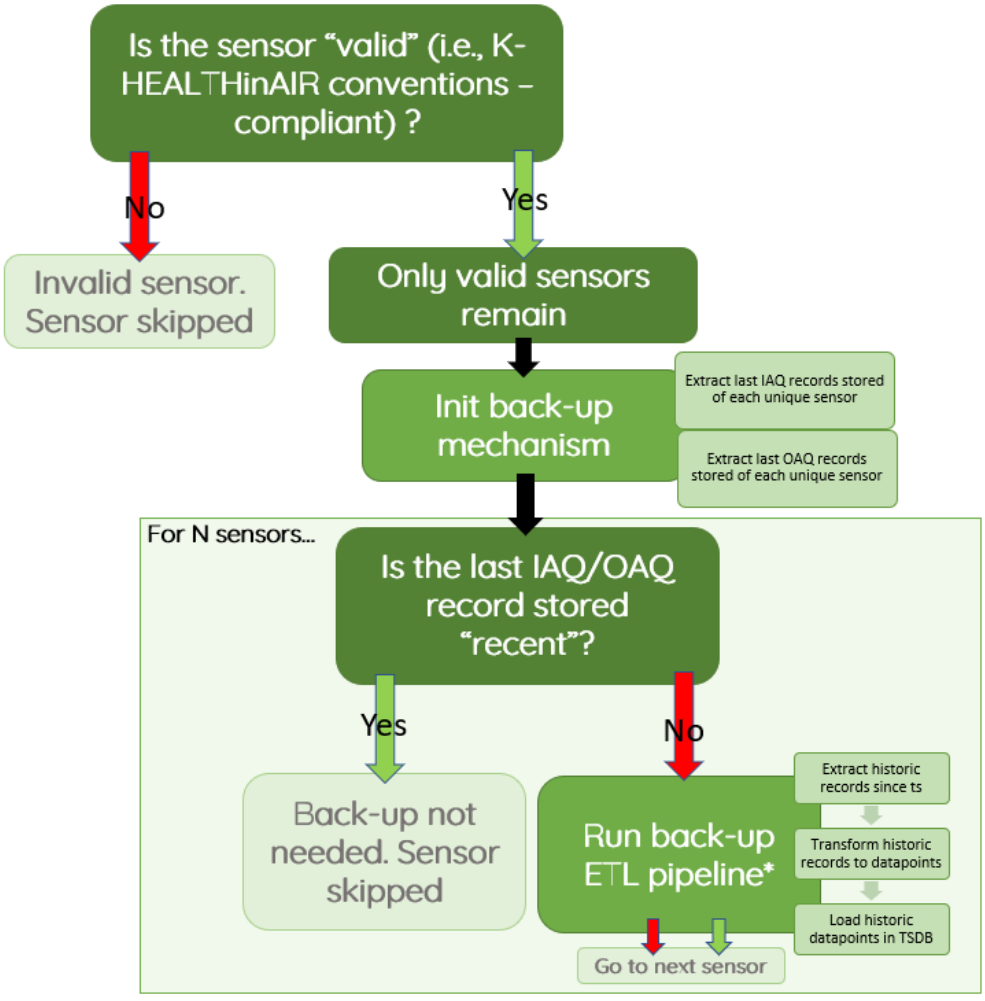

Figure 4: AQDR back-up mechanism schema. (*) For each pipeline separately (IAQ/OAQ).

1) Firstly, the latest stored IAQ and OAQ records (separately) for each distinct sensor in AQDR DB are retrieved, along with their associated timestamps. In this way, the last time the IAQ and OAQ data were ingested is known for each device that has been enrolled at

Funded by the European Union

any point in time. This is performed for all sensors that have at least one datapoint registered in the database. The query to the database is only performed once, and it therefore extracts this information for all the registered sensor devices in a single request.

2) Then, for each valid sensor, AQDR Ingestor checks if back-up is actually needed, depending on if its last timestamp recorded is considered "recent" by the AQDR. This is done separately regarding IAQ and OAQ data (a sensor may need to retrieve past IAQ data, but may not need to retrieve OAQ data, or vice versa).

   The reason for checking this condition is that the back-up for that sensor may not be necessary if the last datapoint recorded is recent enough. The decision is made according to a pre-defined range of time that seems "recent" for AQDR back from the exact moment when the sensor's back-up is being checked. This range of time depends either on if it is an IAQ back-up process or an OAQ back-up process, and in case of IAQ it also depends on the sensor's provider (since they have different scheduled ingestion periodicities, as stated before).

   - If back-up is not needed, that sensor's back-up pipeline is skipped, and the process continues with the next valid one.

   - If back-up is needed for the device, back-up ETL pipeline starts. Historic records of the specific sensor are extracted since the last stored timestamp. Historic responses are afterwards transformed and processed, and the set of historic datapoints is finally ingested into the DB. Hence, the back-up pipeline steps follow a very similar approach to that of the one in the scheduled ETL pipelines. Either the back-up ETL pipeline is successful for a sensor or fails in any of the extract-transform-load steps, the process moves towards the next sensor in the loop. Hence, an unexpected failure in a sensor's back-up does not affect others.

3) Once the entire back-up processes have finished and back-ups for all the valid sensors have been checked or performed, then the scheduled batches of each ETL pipeline (IAQ IB ETL, IAQ MH ETL, OAQ ETL) start running as expected.

IMPORTANT NOTE: In the case of M+H-related requests (i.e., IAQ requests for M+H sensor devices and OAQ requests for all sensor devices in the K-HEALTHinAIR project), a maximum limit of one-month (30 days approximately) datetime range for historic requests to M+H API exists. This response size limit is not handled by the M+H API and seems to be a restriction for making GET requests to the API with a datetime range higher than a predetermined limit. Thus, in order to ensure a valid historic response instead of receiving an *Internal Server Error* that prevents past data to be ingested in our AQDR TSDB, the implemented behaviour of AQDR Ingestor consists of looping over 15-days-length batches/periods for making several requests, until filling in the entire datetime range that is needed for the back-up. This means that if a back-up for more than 1 month is needed for a sensor, instead of requesting historic data since this last timestamp stored (more than 1 month older), the datetime range needed will be previously divided into 15-days periods that will be requested independently (and the historic data of each period will be sequentially ingested until reaching the current timestamp and back-up is thus completed).

Funded by the European Union

2.1.3. AQDR API

Air Quality Data Repository API module consists of an Application Programming Interface (API) that enables to leverage the entire AQDR potential of the K-HEALTHinAIR project. Its main goal is to allow AQ measurements retrieval from our AQDR TSDB, making possible to query data up to several levels of hierarchy (Scenario, Subscenario and Area levels within a Pilot). For these queries to be successful, AQDR DB component must be up and running, and AQDR API must be pointing to the same InfluxDB Organization and Bucket to reach the specific TSDB. *Figure 5* shows the basic AQDR API Swagger UI[8].

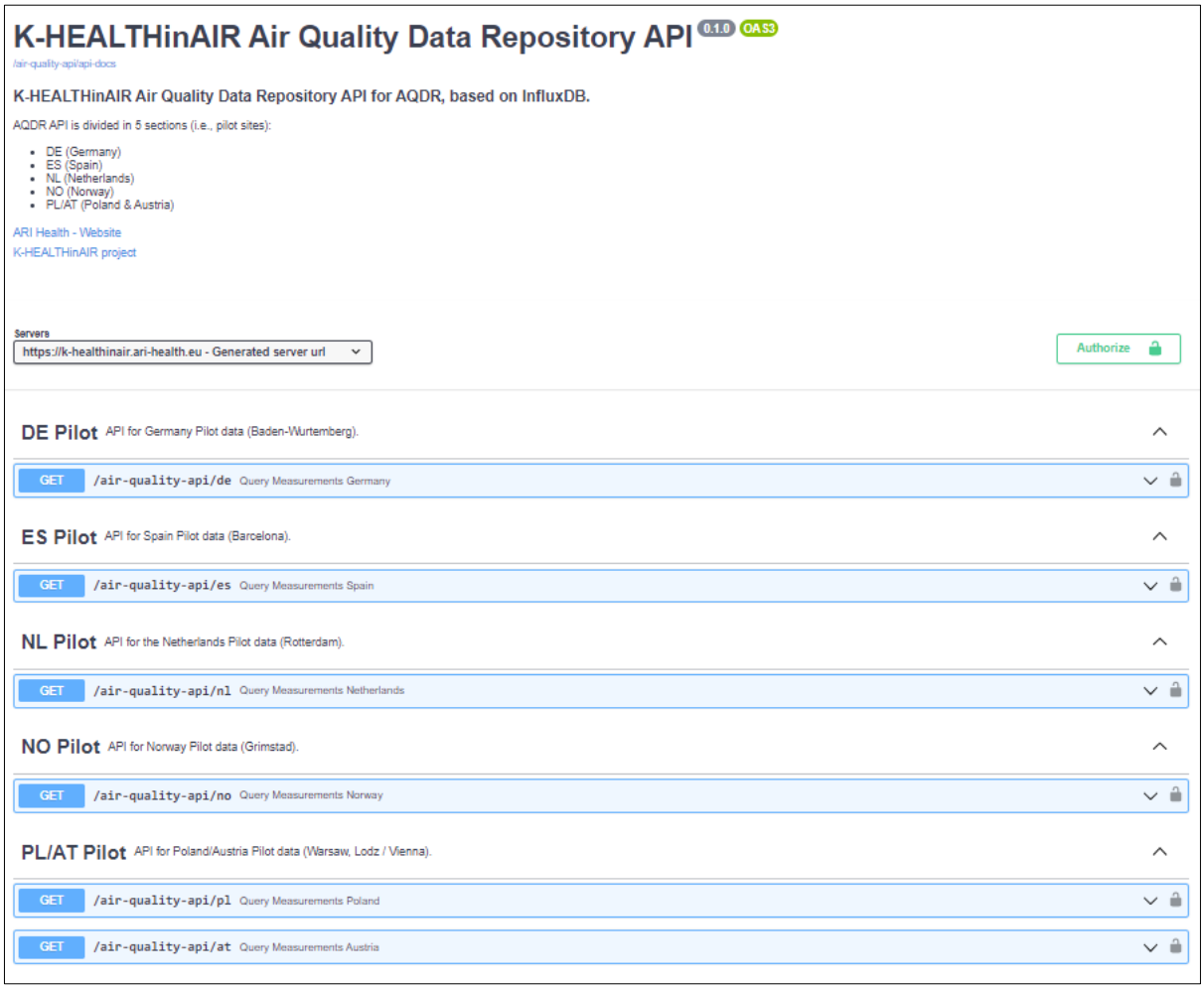

Figure 5: Air Quality Data Repository API Swagger UI.

The API is divided into 5 sections that correspond to the 5 K-HEALTHinAIR pilots (see *Figure 5*): ES Pilot (Spain, Barcelona), DE Pilot (Germany, Baden-Wurttemberg), NL Pilot (the Netherlands, Rotterdam), NO Pilot (Norway, Grimstad) and PL/AT Pilot, which is sub-divided into its two

[8] Swagger, a set of software tools to design and document API RESTful web services (*https://swagger.io)*

Funded by the European Union

locations (PL for Poland, Warsaw and Lodz; and AT for Austria, Vienna). There are GET endpoints dedicated to each particular pilot.

AQDR API is secured based on authentication scheme OAuth 2.0[9] with password and hashing and uses JWT[10] (Json Web Tokens). This means that a valid username and password is needed to be authenticated and therefore having access to K-HEALTHinAIR AQDR TSDB data through AQDR API.

AQDR API offers a powerful capability, which is to allow data retrieval up to different levels of hierarchy, using the pre-established hierarchy tags associated to the sensors data during ingestion processes (i.e., pilot location of the sensor, scenario, subscenario, area). Two file formats are supported for the output files: *.csv* and *.xlsx*. Those files present the requested measurements in a specified time range, where each row is a single datapoint, and the columns provide the following information (see *Figure 6*):

- *_time* (timestamp associated to the datapoint),
- *sensor_id* (ID code of the device of the datapoint),
- IAQ variables of K-HEALTHinAIR Air Quality Data Model (i.e., *co2, humidity, pm1, pm10, pm25, temperature, tvocs, formaldehyde*),
- OAQ variables of K-HEALTHinAIR Air Quality Data Model (i.e., *out_co, out_humidity, out_no2, out_o3, out_pm10, out_so2, out_temperature*),
- *sub_scenario* (optional, depending on the level of detail of the request, i.e., if a specific scenario in a pilot was requested, this column is omitted in the file content because its information is thus already contained in the name of the output file),
- *area* (optional, depending on the level of detail of the request, i.e., if a concrete area in a specific scenario in a pilot was requested, this column is omitted in the file content because its information is thus already contained in the name of the output file).

[9] OAuth 2.0, the industry-standard protocol for authorisation (*https://oauth.net/2)*

[10] JSON Web Tokens, industry standard method for representing claims securely between two parties (*https://jwt.io)*

Funded by the European Union

	_time	sub_scenario	area	sensor_id	co2	humidity	pm1	pm10	pm25	temperature	tvocs	out_co	out_humidity	out_no2	out_o3	out_pm10	out_so2	out_temperature
2	2023-06-01 00:00:00	HOM02	HOM02001	35a9c0825cda4c72974ad52c31602639								0,34	67	2,68	65,28	21,8	0,39	13
3	2023-06-01 00:00:00	HOM05	HOM05005	ab250aa6155042a38fa2997e89c6c8b6								0,15	82	7,41	60,23	16,79	0	14
4	2023-06-01 00:00:00	HOM05	HOM05004	e90c5e64600a4cb2afd88244dabc7064								0,29	82	6	51	25	0	14
5	2023-06-01 00:00:00	HOM05	HOM05002	3178b2ff55a845a7b592fb133c8e2d5d								0,32	67	5	65	15,34	0	13
6	2023-06-01 00:00:00	HOM04	HOM04004	39d5a4df371e461f8a10d9a84209d321								0,16	76	3,73	63,62	21,53	0,71	11
7	2023-06-01 00:00:00	HOM04	HOM04003	060753d3a18e4209a2948823c39dadba								0,21	67	5,37	65	15,38	0	13
8	2023-06-01 00:00:00	HOM04	HOM04002	18137896bcfc4d1a9683e76be53ef39a								0,3	82	6	51	25	0	14
9	2023-06-01 00:00:00	HOM04	HOM04001	a6973c3bf15e4734ac25dfe5c3d3fb61								0,17	67	5,41	65	13,97	0	13
10	2023-06-01 00:00:00	HOM03	HOM03002	e30ea4d0fb64401bb59fc5275fd1143f								0,36	67	2,04	66,39	21,18	0,14	13
11	2023-06-01 00:00:00	HOM03	HOM03001	49870edd785c4fd48cc92fa9f5df200e								0,35	71	2,03	52,37	15,39	0	12
12	2023-06-01 00:00:00	HOM05	HOM05007	511f867529b344718c0aeb449e72e1fc								0,3	82	13	41	26,98	0	14
13	2023-06-01 00:00:00	HOM05	HOM05008	28e689814ca14b6a9a11769305bd99c8								0,17	58	11,65	44,7	20,21	0	14
14	2023-06-01 00:02:00	HOM03	HOM03001	49870edd785c4fd48cc92fa9f5df200e	1118	61	1,8	3	3	22	977							
15	2023-06-01 00:02:00	HOM05	HOM05007	511f867529b344718c0aeb449e72e1fc	439	49	8,4	16	14	21	135							
16	2023-06-01 00:02:00	HOM05	HOM05008	28e689814ca14b6a9a11769305bd99c8	440	43	5,4	9	9	22	1071							
17	2023-06-01 00:02:00	HOM04	HOM04004	39d5a4df371e461f8a10d9a84209d321	594	52	28,2	48	47	20	307							
18	2023-06-01 00:02:00	HOM05	HOM05004	e90c5e64600a4cb2afd88244dabc7064	542	49	6,6	12	11	22	266							
19	2023-06-01 00:02:00	HOM03	HOM03002	e30ea4d0fb64401bb59fc5275fd1143f	737	59	84,6	150	141	20	1859							
20	2023-06-01 00:02:00	HOM04	HOM04003	060753d3a18e4209a2948823c39dadba	540	41	4,2	7	7	23	222							
21	2023-06-01 00:03:00	HOM05	HOM05006	0891f5ae85114fb6a96abc0fcf2355a2	744	47	3,6	7	6	22	552							
22	2023-06-01 00:03:00	HOM04	HOM04002	18137896bcfc4d1a9683e76be53ef39a	561	46	5,4	9	9	24	489							
23	2023-06-01 00:03:00	HOM02	HOM02002	4fd6c927108d4e3d845bdd9ef3bf2b4c	785	58	2,4	4	4	22	531							
24	2023-06-01 00:03:00	HOM05	HOM05005	ab250aa6155042a38fa2997e89c6c8b6	470	42	4,2	7	7	24	307							
25	2023-06-01 00:03:00	HOM02	HOM02001	35a9c0825cda4c72974ad52c31602639	703	52	6,6	11	11	21	1011							
26	2023-06-01 00:07:00	HOM03	HOM03001	49870edd785c4fd48cc92fa9f5df200e	1123	61	1,8	3	3	22	975							
27	2023-06-01 00:07:00	HOM05	HOM05007	511f867529b344718c0aeb449e72e1fc	440	49	8,4	14	14	21	133							
28	2023-06-01 00:07:00	HOM05	HOM05008	28e689814ca14b6a9a11769305bd99c8	440	43	4,2	8	7	22	1073							
29	2023-06-01 00:07:00	HOM04	HOM04004	39d5a4df371e461f8a10d9a84209d321	594	52	27,6	48	46	20	307							
30	2023-06-01 00:07:00	HOM03	HOM03002	e30ea4d0fb64401bb59fc5275fd1143f	732	59	83,4	147	139	20	1778							

Figure 6: Example of Excel output file from AQDR API for request of all HOM in AT.

2.2. Discrete Air Quality Sampling Repository (DAQSR)

The Discrete Air Quality Sampling Repository (DAQSR) refers to the module that stores all the discrete measurements periodically sampled during the duration of the K-HEALTHinAIR project. The considered samplings are:

- Volatile Organic Compounds (VOC)
- Passive Sampling of VOCs (VOP)
- Particles Matter (PMX)
- Microbiome culture data (MCC)
- Microbiome sequencing data (MCS)
- Formaldehydes (FDH)
- Polycyclic Aromatic Hydrocarbons (PAH)
- Radon (RAD)

The kind of sampling and its periodicity is defined in each project pilot. Each of these discrete samples will be associated to a pre-established identification code, which is explained in Discrete Air Quality Samplings Identification.

Due to the different nature of the discrete samplings, not only the content information, but also file sizes or even file formats could differ, so there is no single solution for their storage that fits the requirements of each discrete sampling type. Because of this reason, DAQSR is based on MinIO[11], a High-Performance Object Storage. MinIO is an AWS S3 (i.e., Amazon Simple Storage Service) compatible object storage that can handle large objects (up to 5TB) and even unstructured data. It was initially built for large scale AI/ML, data lake and database workloads, so it fits perfectly with the context of the K-HEALTHinAIR project. It also provides MinIO Client Software Development Kit (SDK), which offers an API to access any S3 compatible object

[11] MinIO, High Performance Object Storage for AI (*https://min.io*)

Funded by the European Union

storage server and is useful for data analysis purposes. Moreover, it is hardware-agnostic, which fits with our microservices-oriented architecture approach too.

For K-HEALTHinAIR DAQSR, a proposed bucket and folder structure to store discrete samplings is presented here:

- One bucket per K-HEALTHinAIR pilot, that will contain all the discrete air quality sampling and other data related to these measurements (see *Figure 7*).
- In each bucket, one parent folder in the root bucket directory for each scenario (see *Figure 8*). Each of these folders will host all the files of the discrete measurements that have been carried out in the different scenarios within the pilot, since these files could be easily identified by their filename (obeying K-HEALTHinAIR ID Convention). This structure proposal is preliminary and could change depending on the progress of the implementation.

This is only a first proposal for a preliminary structure to organise the information within each pilot, but it can also be changed throughout the project as needs dictate.

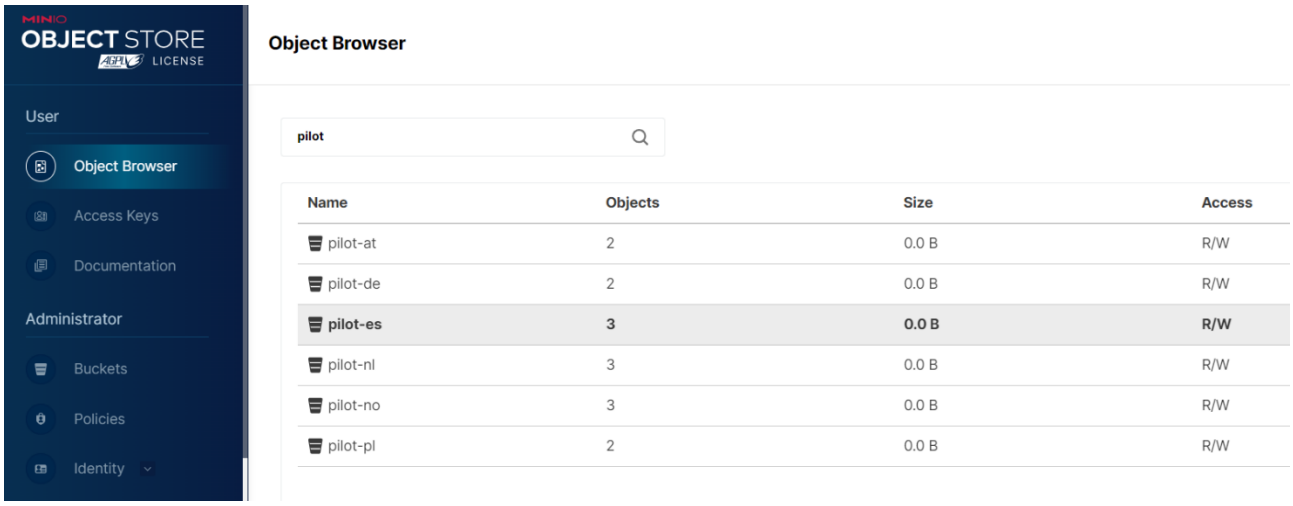

Figure 7: DAQSR main bucket structure.

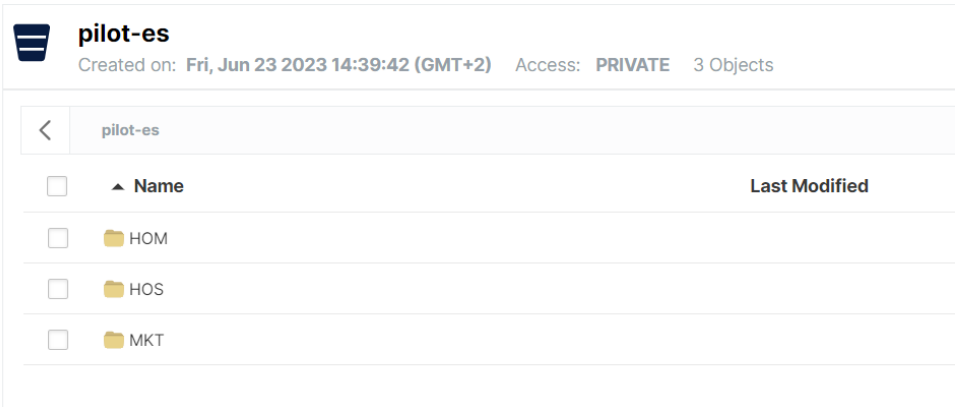

Figure 8: DAQSR pilot folder structure within a certain bucket.

Funded by the European Union

2.3. Health Data Hub (HDH)

The Health Data Hub (HDH) is the component of the platform that will be responsible for clinical data storage and management (*Figure 9*). As mentioned before, it will include:

- Clinical Data Repository (CDR), that will be based on HAPI FHIR. This stands for a complete Java implementation of the HL7 FHIR standard for healthcare interoperability. In this way, medical information stored in K-HEALTHinAIR CDR according to HL7 FHIR specifications will be interoperable.

- Clinical Data Repository API (CDR API), which is a RESTful API that wraps native CDR Database.

Since the clinical partners have not yet defined the type and format of the patient data to be collected, the HDH component of the K-HEALTHinAIR Platform is currently under definition.

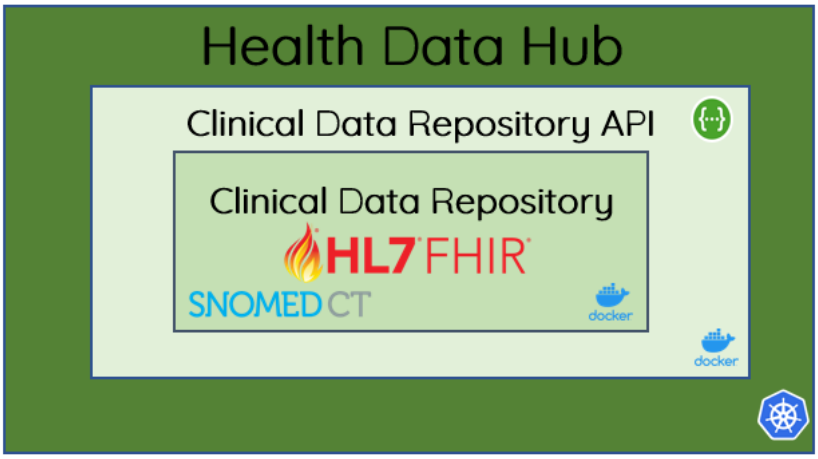

Figure 9: HDH modules: CDR and CDR API.

Funded by the European Union

3. K-HEALTHinAIR Platform conventions

This section presents the K-HEALTHinAIR Platform ID conventions and abbreviations that are common for all the pilots of the project, and the Air Quality Data Model.

During these first 12 months of the project lifecycle, several conventions were defined together with the consortium to harmonise the information (IAQ/OAQ parameters naming and units, discrete sampling measurements naming and formats, participants identification, sensor devices tagging…) from the different pilots. Each pilot coordinator has defined the best naming approach for all the locations in his/her pilot, that could be used from here on out to uniquely identify those locations in the K-HEALTHinAIR project .

The K-HEALTHinAIR ID Convention plays a crucial role in entire K-HEALTHinAIR data management, processing and analysis. Since the agreement of these codes, each specific area or location that is considered in the project should be associated to a unique ID. This helps to:

1) link sensors and samples to a specific area,
2) match participants with their associated area, and
3) uniformly identify data and insights quickly, associating them to their correspondent pilot/scenario/area.

These codes will be kept during the entire project lifecycle.

3.1. K-HEALTHinAIR ID Convention

3.1.1. General K-HEALTHinAIR abbreviations

Several abbreviations have been adopted by the entire consortium, from the higher level of experimentation (i.e., project pilots) to the more detailed one (i.e., areas or locations within a certain scenario in a pilot). The aim is to recognise in an easy, quick, and harmonised way each sensor device, sample, or participant through an associated code that includes information about the specific location which it should be linked with.

To this end, some general abbreviations have been defined:

- Pilots. Although at the beginning of the project 5 pilot sites were described, due to the different nature of the experiments and regulatory aspects, K-HEALTHinAIR ID Convention considers 6 pilots (Austria and Poland Pilot has been split). The pilot code is depicted as a fixed 2-letter code (see *Table 1*).

- Scenarios. The scenario code is depicted as a fixed 3-letter code (i.e., the scenario type, see *Table 1*) followed by a 2-digit code (01-99, i.e., different buildings or protocols of the same type).

- Areas. Depending on the pilot and scenario, some spaces or specific locations within scenarios could be differentiated too. The area code, a 3-digit code (001-999, see *Table*

Funded by the European Union

1) enables listing several spaces within a scenario (e.g., different classrooms in a certain school, or different working areas in a hospital).

Table 1 K-HEALTHinAIR ID Convention for pilots, scenarios, and areas of the project.

PILOT CODE (ISO 3166-1[12])		SCENARIO CODE			AREA CODE
		TYPE		#	#
AT	Austria	CAN	Canteen	01-99	001-999
DE	Germany	LEC	Lecture Hall		
ES	Spain	HOM	Home		
NL	The Netherlands	HOS	Hospital		
NO	Norway	MET	Metro Station		
PL	Poland	MKT	Market		
		RES	Residence Home		
		RET	Retirement Home		
		SCH	School		

For further clarification of the ID logic, a few examples are presented here:

- *ESHOS01001* could represent a specific location (area 001, e.g., a waiting room) in a specific hospital (HOS01) in Spain pilot (ES).

- *PLSCH02007* could represent a specific classroom (area 007, if several rooms monitored in the scenario SCH02) in the school number 2 (SCH02, if several schools in the pilot) of Poland pilot (PL).

- *ATHOM03004* could represent a specific home (area 004, if several homes in the scenario HOM03) in the protocol number 3 of homes scenario (HOM03, e.g., groups according to the heating system, if several homes' protocols in the pilot) of Austria pilot (AT).

3.1.2. Sensor Devices Identification

According to the pre-established codes in *Table 1* for pilot, scenarios and areas, each sensor device (either from INB or M+H) must be linked to a specific area within the project, in order to be able to locate where its AQ measurements are being recorded. Thus, in addition to the unique, default device ID provided by INB or M+H, K-HEALTHinAIR AQDR attaches auxiliary tags to each sensor to locate the device in the context of the project.

[12] ISO 3166-1, an international standard defining codes for the names of countries (*https://en.wikipedia.org/wiki/ISO_3166-1*)

Funded by the European Union

All AQ datapoints that are recorded in AQDR DB are therefore ingested together with:

- *pilot* tag, that specifies in which one of the 6 valid pilots (see *Table 1*) is the device located (e.g., DE).

And several incremental tags:

- *scenario* tag, that indicates in which of the 9 valid scenario types (see *Table 1*) is the device located (e.g., CAN).
- *sub_scenario* tag, that represents the specific scenario or building of a certain type within a pilot, i.e., concatenates given *scenario* tag with 2-letter numbering code (e.g., CAN01 or CAN02 if there are more than two canteens in same pilot).
- *area* tag, that represents a concrete space within a certain subscenario, i.e., concatenates given sub_scenario tag (e.g., CAN01) with 3-letter numbering code (e.g., CAN01001 represents Area 001 -kitchen area- within CAN01).

The AQDR sensors annotation approach for each sensor device enrolled in K-HEALTHinAIR Platform allows to easily extract a set of devices that fulfil similar conditions or perform aggregations depending on the desired level of detail of the data analysis studies. For example, within a certain pilot, records from all sensor devices located in all types of homes could be aggregated (since these records carry a tag *scenario=HOM*); but they could also be aggregated in smaller groups thanks to their specific tag *sub_scenario={HOM01, HOM02, HOM03…}* according to different heating systems; or even so in a more detailed level, considering as independent each of the areas by their tag *area={HOM01001, HOM01002, HOM02001, HOM02002…}*. and aggregating the sensor devices that are tagged as located in the same space.

- Sensors Description File

The AQDR Ingestor and AQDR DB follow some fixed guidelines and rules about the enrolment, tagging and ingestion of sensor devices to make all this explained identification possible in K-HEALTHinAIR Platform. To this end, ATOS (as AQDR developer) and M+H and INB (as sensor devices providers) have agreed a fixed format to respectively receive and include sensors information.

As described before, every sensor device in the K-HEALTHinAIR project should include several tags that contain important information about its precise location. This data will be used to tag each datapoint ingested into the database and will be supplied by sensor providers during the devices installation process. The so-called *Sensors Description File* is the csv/xlsx file where all the information about the sensor devices that participate in the IAQ/OAQ continuous monitoring is maintained and periodically updated. It is collaboratively maintained in a secured folder within ATOS SharePoint that has been shared with M+H and INB.

This *Sensors Description File* contains certain columns that indicate the necessary information for each specific device (i.e., row in the file):

Funded by the European Union

- *SENSOR_ID*: the original ID of the sensor device.
- *APIKEY*: the associated key that allows to perform IAQ GET requests through the providers' APIs.
- *PROVIDER*: the code that identifies the sensor's provider (only two valid values: {IB, MH}, where IB=InBiot, MH=Mann+Hummel).
- *PILOT*: the code that identifies the pilot where the device is (only six valid values: {AT, DE, ES, NL, NO, PL} according to *Table 1*).
- *SCENARIO*: the code that identifies the specific scenario within the pilot. It must consist of a valid three-letter code that represents the scenario type, followed by a two-digit code that numbers the concrete scenario where the device is (see *Table 1*).
- *AREA*: the code that identifies the specific area within a certain scenario. It must be a number in the range 1-999.
- *OAQ_APIKEY*: the associated key that allows to perform OAQ GET requests through the provided building number.
- *BUILDING*: the number of the building (for OAQ retrieval) where the sensor device is theoretically located.

This BUILDING column is mandatory for OAQ data retrieval. Each unique building encompasses a set of sensors that are near enough to assume they will have the same OAQ record at a given timestamp. This explains a special feature of OAQ ETL Pipeline in AQDR Ingestor: for each scheduled iteration of the OAQ ETL Pipeline, the Extract step is only performed once for each unique building (i.e., only one GET request per building), and the OAQ record of the building is temporarily stored in cache, so all sensor devices that belong to it can quickly recover the record instead of making the same GET request again unnecessarily. Thereby, this cache with OAQ records is cleaned after each ETL batch iteration scheduled.

The already mentioned validation processes performed by AQDR Ingestor (*2.1.2 AQDR Ingestor*) are indeed based on these specifications. This means that a sensor device is considered in the ETL pipelines only if it is compliant with all these conventions, i.e., if their supplied values for the columns fall into the valid ranges of values for each category (e.g., PROVIDER must be IB or MH, AREA must be a number between 1-999, etc.). Sensor providers and pilot leaders are responsible for including the correct information in this file, so it must be carefully revised, since for instance introducing an incorrect area for a device will cause to ingest fake or corrupt data in AQDR DB.

3.1.3. Discrete Air Quality Sampling Identification

Preestablished codes for discrete air quality samplings were also defined. Each sampling type has been linked with a fixed 3-letter code (see *Table 2* below).

Funded by the European Union

Table 2: K-HEALTHinAIR Discrete Air Quality Sampling types 3-letter codes.

DISCRETE AIR QUALITY SAMPLING TYPE CODE	
Volatile Organic Compounds	VOC
Volatile Organic Compounds (Passive measurement)	VOP
Particles Matter	PMX
Polycyclic Aromatic Hydrocarbons	PAH
Microbiome (Culture)	MCC
Microbiome (Sequencing)	MCS
Radon	RAD
Formaldehyde	FDH

Since all these measurements are always going to be performed in a certain area, their complete code should also include a prefix that indicates the precise location where the samplings were conducted. Besides, it should contain the date and time (ISO 8601[13] format) when the sampling started or was conducted. *Table 3* below shows the expected code for DAQS.

Table 3: Complete 13-digit code (plus datetime) identification code for DAQS.

PILOT CODE	SCENARIO CODE	AREA CODE	DAQS CODE	DATE + TIME (ISO 8601)
{AT, DE, ES, NL, NO, PL}	{CAN, LEC, HOM, HOS, MET, MKT, RES, RET, SCH} + (01-99)	(000-999)	{VOC, VOP, PMX, PAH, MCC, MCS, RAD, FDH}	YYYY-MM-DD-HHMM

Some additional notes about this convention:

- If the sampling has not been carried out in a specific date, but during several days, it has been agreed that the sampling identification code will include the start datetime and then the exact dates will be included in the file content.

- Area code 000 is allowed in complete discrete samplings code. This '000' code is reserved for those samplings whose associated area is not any specific area within the scenario, maybe because they were carried out outside the scenario. For example, it could happen in a scenario where there are 20 indoor areas (then Area codes= [001, 020]), but a

[13] ISO 8601, an international standard for date and time-related data (*https://en.wikipedia.org/wiki/ISO_8601*)

Funded by the European Union

specific discrete sampling was decided to be performed outside of the scenario (e.g., in the entrance of the metro station or in the main entrance of the hospital).

3.1.4. Participants Identification

Each participant in the K-HEALTHinAIR project must be uniquely identified within the context of the project. To this end, this subsection shows the pre-established conventions to define the guidelines to name a specific participant. To maintain participants' anonymity and preserve their privacy, the proposed identification code only refers to each participant with a 3-digit code (001-999) that is unique in their pilot and therefore is not restarted within a certain pilot.

In addition, in the same way as in the discrete samplings, each participant-specific code should also include a prefix that refers to the information about the location of the project which the participant is linked with. As baseline rule, each participant in the K-HEALTHinAIR project must be linked to one area at maximum.

In other words, it is enough to represent a participant in a pilot through his/her participant code (see *Table 4* below), since this 3-digit code will be unique for each participant in a pilot. In this way, all participants could be identified even if their associated main area changes over time during the project lifecycle, which could be the case in some pilots (NL, NO) through these last 3 digits. This enables some flexibility for those more unpredictable events. Finally, in the entire 13-digit code that represents a participant, the location code (i.e., scenario code plus area code) is complementary information that could be very useful to directly match participants with AQ data from sensor devices or discrete measurements associated to those locations.

Table 4: Complete 13-digit code identification code for participants.

PILOT CODE	SCENARIO CODE	AREA CODE	PARTICIPANT CODE
{AT, DE, ES, NL, NO, PL}	{CAN, LEC, HOM, HOS, MET, MKT, RES, RET, SCH} + (01-99)	(000-999)	001-999*

Some additional notes about this convention:

- Participant code is not restarted within a pilot, and it is thus unique for each participant in his/her pilot context. For example, if there are 30 participants associated to scenario HOM01 and 20 participants associated to scenario HOM02 in the same pilot, numbering of participants in HOM02 (031-050) starts after identifying all participants in HOM01 (001-030).

- Thus, a participant in a pilot (e.g., participant *018* in NL) will be uniquely identified in the context of its pilot. In this way, unpredictable changes in participant's associated area could be performed during the project (e.g., the entire participant code for participant 018 could be NLHOS01001*018* if linked with the area HOS01001 during the first year, and

Funded by the European Union

then it could be changed to NLRES02002*018* if linked with the area RES02002 in the second year). Nevertheless, these are mostly exceptions in the project.

- Area code 000 is allowed in complete participants code. This '000' code is reserved for those participants that are not associated to any specific area within the scenario, but to the entire scenario. For example, this could happen in a scenario where the staff is continuously working in different named areas and therefore could not be linked with only one of them. It was pilot leaders' task to decide and consider in certain cases if the participant was going to spend enough time in a concrete area to be linked with it, or if on the contrary this was not clear enough and therefore the safer decision was to link him or her with the whole scenario.

3.2. K-HEALTHinAIR Air Quality Data Model (AQDM)

The so-called Air Quality Data Model (AQDM) is an agreement between AQ sensor devices providers (InBiot and Mann+Hummel), K-HEALTHinAIR Platform technical partner (ATOS), and also the rest of the K-HEALTHinAIR project partners, that stands for a Common Data Model for IAQ and OAQ parameters. Its main objective is to harmonise the nomenclature for all the AQ variables collected and continuously monitored by the sensor devices, and afterwards stored in AQDR Database as such.

Table 5 and *Table 6* respectively show Indoor and Outdoor AQDM. Original, raw parameter names in each platform, together with the sensors' limits of detection and measurement units, are presented, as well as the global K-HEALTHinAIR AQDM parameter name and type of value that is stored in AQDR TSDB.

Funded by the European Union

Table 5: Indoor Air Quality Data Model.

Indoor AQDM				
AQDM Parameter	Unit	AQDM Type	INB Parameter [min, max]	M+H Parameter [min, max]
temperature	°C	Float (.2)	temperature [-40, 145]	t [-20, 100]
humidity	%		humidity [0, 100]	h [0, 99]
tvocs	ppb		vocs [0, 60000]	voc_ppb [0, 60000]
pm1	ppb		pm1 [0, 5000]	NA
pm10	µg/m³		pm10 [0, 5000]	pmten [1, 1000]
pm25	µg/m³		pm25 [0, 5000]	pmtwo [1, 1000]
co2	ppm		co2 [0, 10000]	cotwo [400, 10000]
formaldehyde	µg/m³		formaldehyde [0, 1250]	NA
°C: Celsius degrees, %: relative humidity percentage, ppb: parts per billion, ppm: parts per million, µg/m³: micrograms per cubic meter				

Table 6: Outdoor Air Quality Data Model.

Outdoor AQDM			
AQDM Parameter	Unit	AQDM Type	Outdoor Parameter (M+H – AerisWeather)
out_temperature	°C	Float (.2)	t
out_humidity	%		h
out_pm25	µg/m³		pm2.5
out_pm10	µg/m³		pm10
out_co	ppb		co
out_no2	ppb		no2
out_so2	ppb		so2
out_o3	µg/m³		o3
°C: Celsius degrees, %: relative humidity percentage, ppb: parts per billion, µg/m³: micrograms per cubic meter			

Funded by the European Union

4. K-HEALTHinAIR Platform components

The platform is made up of the following components, accessible through the links provided below. All these modules are integrated thanks to the underlying microservices paradigm that has been followed during their development, which ensures each component is functional by itself.

- Air Quality Data Repository Time-Series Database (AQDR TSDB):

https://air-quality-repository.k-healthinair.ari-health.eu

- Air Quality Data Repository API (AQDR API):

https://k-healthinair.ari-health.eu/air-quality-api

- Discrete Air Quality Samplings Repository (DAQSR):

https://discrete-air-quality-repository.k-healthinair.ari-health.eu

Funded by the European Union

5. K-HEALTHinAIR AQDR User Guide

This section goes through the main workflows of AQDR and mimics a possible user behaviour in the K-HEALTHinAIR Platform environment. Firstly, a basic explanation about the user capabilities in AQDR TSDB is presented. Then, some additional comments about AQDR API correct usage are provided to easily leverage its potential and get AQ data.

Since AQDR DB is secured, a general user (together with its associated password) will be provided with only read permissions to allow consortium to enter the application and access InfluxDB UI and dashboards, see continuously monitored IAQ and OAQ data, etc. (see *Figure 10*).

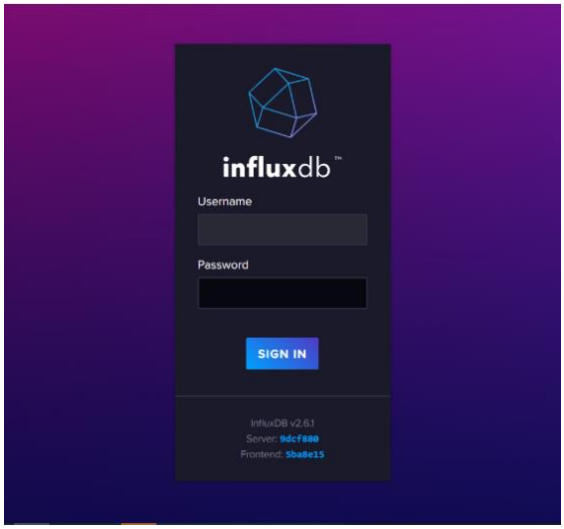

Figure 10: Login page for AQDR Database.

For plain users' purposes, the most useful component is the Data Explorer (see *Figure 11* and *Figure 12*). This module consists of a user-friendly interface that shows the main graph in the upper part, where the progress over time of IAQ/OAQ data can be observed. Besides, it is possible to perform queries interactively, so only the desired data is visualised, thanks to several provided filters. Within a selected bucket, it is possible to query and filter data according to the measurement and, which is of more interest for K-HEALTHinAIR project, according to different levels of hierarchy thanks to tags.

A wide range of personalised queries could be performed thanks to the filters the UI provides (see *Figure 11*). It is possible to filter by different tags (by pilot, scenario, subscenario, area, sensor_id and IAQ/OAQ measurements), and filter only desired fields (temperature, humidity, out_co2, etc.). Moreover, filtering by dates is allowed (even selecting a custom time range). It is feasible to see raw data stored in the TSDB, to customise the visualisation of the data (graph, heatmap, histogram, mosaic, etc.) or to download the raw data as .csv file too.

Funded by the European Union

Figure 11: AQDR DB UI of its main component, Data Explorer.

Figure 12: AQDR DB Data Explorer UI datapoint pre-visualisation.

For its part, AQDR API is already quite documented, so every user that would like to leverage AQDR data through this module will be able to perform successful queries following the provided guidelines. For each GET method, there are several required parameters and other optional parameters (see *Figure 13*). AQDR API Swagger UI also offer some helpful errors during development and tests (see *Figure 14*).

- Pilot (required), which refers to the specific pilot within the project from where the IAQ/OAQ records are desired. It must be compliant with K-HEALTHinAIR Pilots ID convention of *Table 1*. AQDR API Swagger UI already splits the pilots in sections so this parameter should not be specified if the request is being performed from the UI.
- *file_ format* (required), which refers to the desired output file format. Supported values are only "csv" (.csv file format) and "xlsx" (.xlsx file format). AQDR API Swagger UI already offers a dropdown with the allowed values to avoid passing invalid file format values in the request.
- *scenario* (required), which refers to a scenario type within a pilot from where the IAQ/OAQ records are desired. It must be compliant with K-HEALTHinAIR Scenarios ID

Funded by the European Union

convention of Table 1. AQDR API Swagger UI already offers a dropdown with the allowed values to avoid passing invalid file format values in the request, so these dropdowns are specific to the pilot/section.

- *sub_scenario* (optional), which refers to the specific scenario within a pilot from where the IAQ/OAQ records are desired. Here, only digits in the range 01-99 are valid or a 400 Bad Request code will be returned.
- *area* (optional), which refers to the specific area within a concrete scenario from where the IAQ/OAQ records are desired. Here, only digits in the range 001-999 are valid or a 400 Bad Request code will be returned.
- *start_datetime* (optional), which refers to the specific datetime from when the data is queried to AQDR DB. It must be in ISO 8601 format. By default, it is the first expected day for the sensors to be running in the context of the project (2023-05-01).
- *end_datetime* (optional), which refers to the specific datetime from when the data is queried to AQDR DB. It must be in ISO 8601 format.

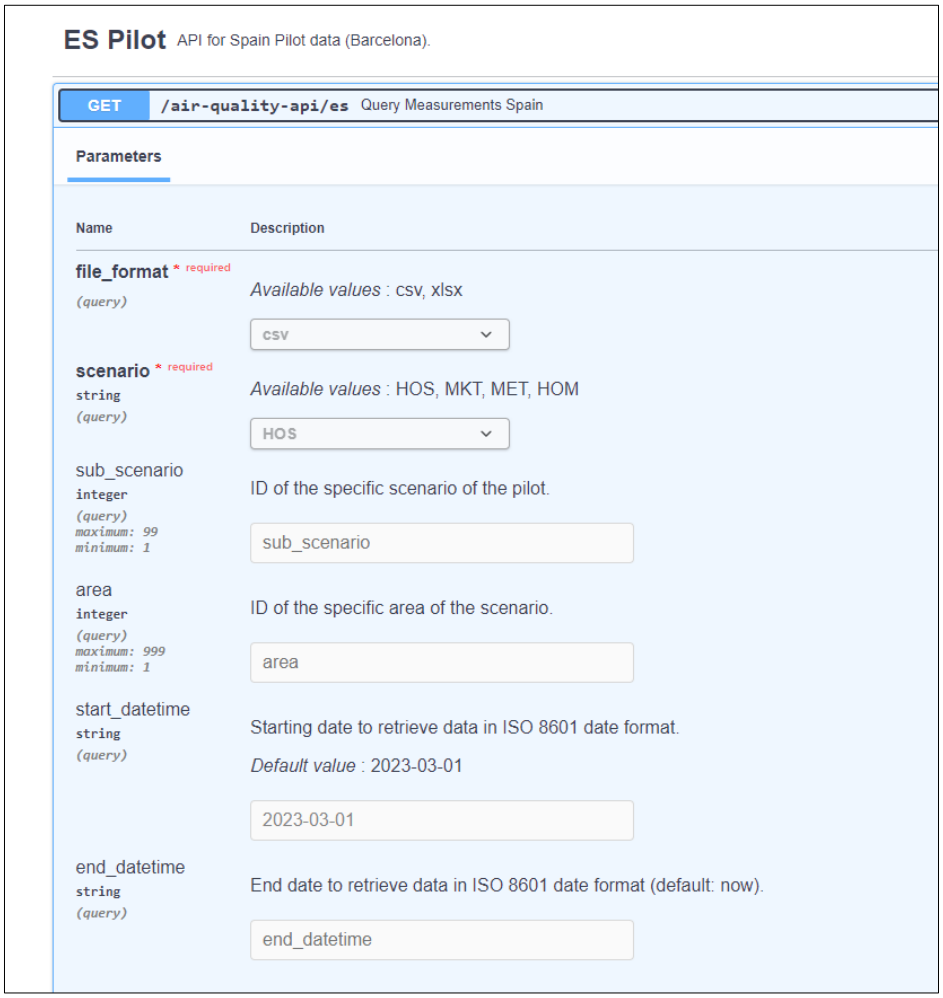

Figure 13: AQDR API GET method specifications: endpoint, parameters, etc.

Funded by the European Union

Figure 14: Example of an AQDR API Swagger UI bad request error.

As described before in AQDR API specifications, the endpoints are secured so a valid username and password is required to be authenticated and thus being able to access K-HEALTHinAIR continuous monitoring AQ data from AQDR TSDB. To this end, the username and password should be provided by the user through a form that is displayed if "Authorize" green button is clicked (see Figure 15 below).

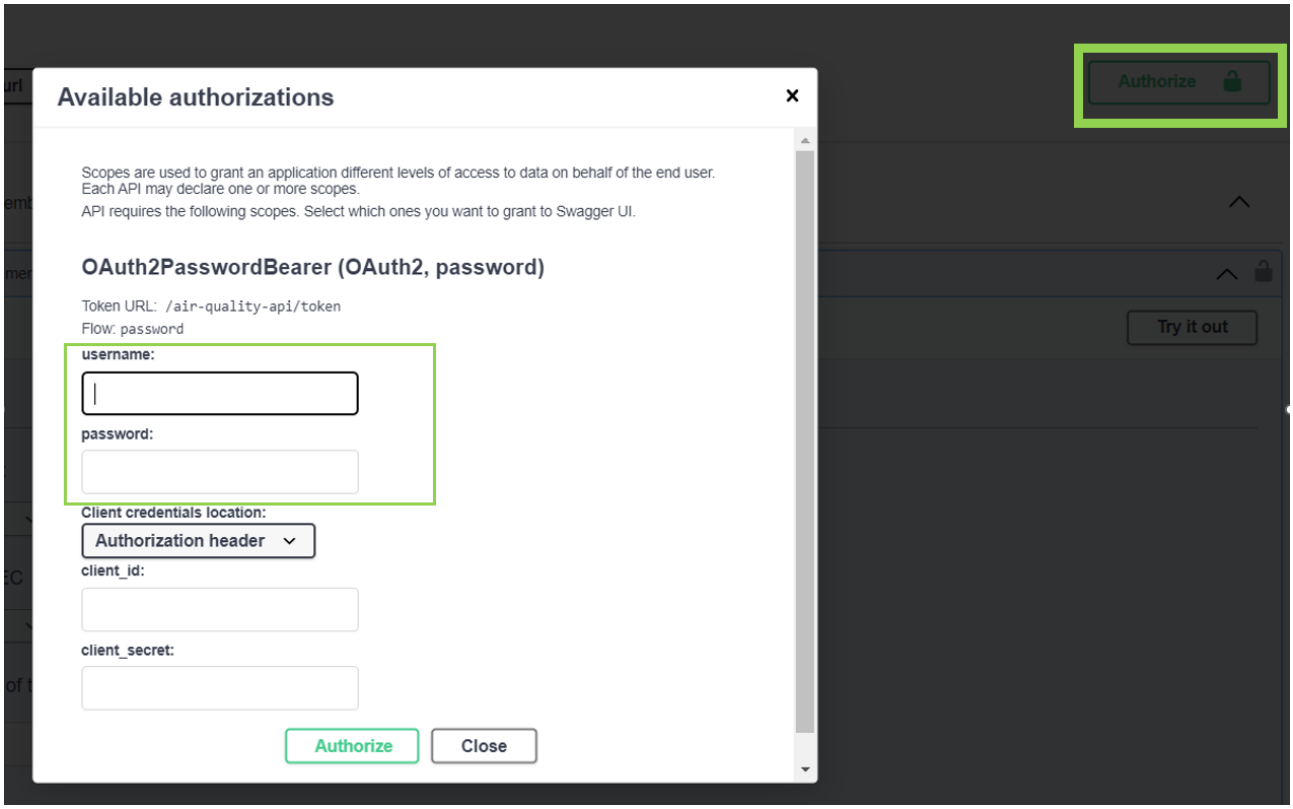

Figure 15: OAuth2 form of secured AQDR API.

AQDR API will inform the user when the .csv/.xlsx requested file is ready for download. The filename of the file is generated according to the request in a format "PPSSSssaaa_YYYY-MM-DD_YYYY-MM-DD", being PP=2-letter code for the pilot, SSS=3-letter code for the scenario type, ss=2-digit code for the specific scenario (optional, depending on the request), aaa=3-digit code for the concrete area in the scenario (optional, depending on the request), and YYYY-MM-DD the dates of start and end datetime requested, respectively.

It is important to emphasise that this hierarchical query approach allows to retrieve IAQ/OAQ information until different levels of detail. This is the reason why *sub_scenario* and *area* parameters are optional, and their associated columns could therefore be missing in the output files to avoid redundancy.

Funded by the European Union

Here a possible use case is presented to show the wide range of options this approach offers to the data analysts. In the case of Spanish pilot in Barcelona, we could aggregate data until different levels of hierarchy:

- At scenario level (higher and most general level), for example, to compare AQ data of all enrolled homes in ES pilot with AQ data in a different scenario (e.g., the hospital). Then, *sub_scenario* and *area* tags will not be specified in the request, so information of all HOM subscenarios (HOM01, HOM02 and HOM03 equally) will be obtained.

- At subscenario level, for example, to focus only on a certain group of houses in HOM scenario of ES pilot. To this end, it is enough to select an existent *sub_scenario* (i.e., the digit that identifies a concrete subscenario). Then, *sub_scenario* column will not appear in the output file, since it will be contained in the filename and all datapoints obtained will come from the specified subscenario (e.g., HOM01 houses only) and would carry the same *sub_scenario* column value (which would not offer any valuable information in the dataset).

- And even more, at area level, for example, to focus the analysis on a concrete house that could be of interest (e.g., HOM01001). To this end, it is necessary to specify an existent area (i.e., "001" in this example) within a scenario. For this reason, it is not possible to select a specific area without having a specific scenario previously selected. Then, neither *sub_scenario* nor *area* columns will appear in the output file for the same reason as in the previous case. This information will already be contained in the filename, and all datapoints in the file content would carry the same *sub_scenario* and *area* columns values (which could be redundant).

Funded by the European Union

6. Conclusions

In conclusion, this deliverable D4.1: K-HEALTHinAIR BigData Platform (Version I) details the first prototype of the K-HEALTHinAIR Platform at month 12 of the project. For this first version of the platform, AQDR module is the most mature component, and it is currently working and registering real AQ data from several scenarios of the project. On the contrary, HDH modules (CDR and CDR API) are still under definition, pending the elucidation of the clinical data to be gathered.

Due to its unavoidable dependencies on other WP1 tasks running in parallel, some definitions will be finished during the following months.

K-HEALTHinAIR Platform will be exploited by the data analysis tasks to be carried out in the next months. This will allow to gather the feedback of real users and propose new updates.

For this reason, several future lines of work and next steps are proposed here:

- First consultation activities for the K-HEALTHinAIR Open Access Platform with the external stakeholders under WP4, in addition to the workshop of the platform overview planned for 3rd General Assembly meeting in Vienna (M13).

- Discuss and agree with Hospital Clinic de Barcelona the details about periodic exports of clinical data and questionnaires stored in REDCap[14].

- Discuss and agree a Clinical Data Model (CDM) where all clinical variables and clinical data is defined, in the same way as an Air Quality Data Model has been closed. Our HDH will use a well-known common terminology (e.g., SNOMED-CT[15]).

- Build all the necessary components of the Health Data Hub, i.e., Clinical Data Repository which is based on HL7 FHIR standard and Clinical Data Repository API.

[14] REDCap, a secure web application for building and managing online surveys and databases. (*https://www.project-redcap.org*)

[15] SNOMED-CT, the most comprehensive, multilingual, accurate and important clinical terminology (*https://www.snomed.org*)

Funded by the European Union