MISSIONE 4
ISTRUZIONE
RICERCA

ITINERIS

Completion of three open-air terrestrial platform repositories (FO3X, Durum FACE, Ndep) for automatization, near realtime transfer and long-term storage of data, and transfer of harmonized data from the repositories to the project hub.

Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA

Deliverable number:	D6.18
Work package:	WP6.03
Intermediate Objective:	IO6.7
Deliverable type:	X Document, report
	☐ Websites, patent filings, videos, etc.
	☐ Other: please specify ....
Dissemination level:	X Public
	☐ Restricted
Estimated delivery (bimester):	B14
Actual delivery date:	25/02/2025
Author(s) (Partner-OU):	Elena Paoletti (CNR-IRETFI), Montaghi Alessandro (CNR-IRETFI), Elena Marra (CNR-IRETFI)
Reviewed by:	ITINERIS Executive Board
Note:	

IR0000032 – ITINERIS, Italian Integrated Environmental Research Infrastructures System - CUP B53C22002150006 (D.D. n. 130/2022)
Funded by EU - Next Generation EU
Mission 4 “Education and Research” - Component 2: “From research to business” -
Investment 3.1: “Fund for the realisation of an integrated system of research and innovation infrastructures”

Table of contents

1. INTRODUCTION: ITINERIS – FREE-AIR MANIPULATIVE TERRESTRIAL ECOSYSTEM PLATFORMS METADATA REPOSITORIES

ITINERIS (Italian Integrated Environmental Research Infrastructures System) is a broad-scale and long-term project that aims to build the Italian Hub of Research Infrastructures in the environmental scientific domains, to provide multi-disciplinary knowledge, predict how ecosystems will respond to future global changes and formulate strategies to counteract and mitigate such changes. The main goal of WP6.03 is to promote data integration and harmonization to allow data automatization and digitalization as prerequisites for free access to facilities, data services and research outputs. Deliverable D6.18 is part of the work carried out within ITINERIS WP6.3, which specifically regards the “Harmonization and integration of free-air manipulative platforms data in terrestrial ecosystems”.

The ANAEE (Analysis and Experimentation on Ecosystems) Research Infrastructure consists of long-term, field-scale experimental manipulation platforms located across various terrestrial ecosystems and climatic conditions to have a comprehensive understanding of the fate of forest ecosystems and crops due to elevated Carbon dioxide, Tropospheric Ozone and Nitrogen deposition in a long-term perspective. Data integration, harmonization and data sharing are essential to exploit data potential from the different manipulative experiments for contrasting the effects of climate change drivers (i.e. elevated nitrogen, CO2and ozone), developing realistic risk assessments and models on the responses of terrestrial ecosystems to global changes by overcoming the limits of having standalone experimental platforms.

The goal of having one access point to the knowledge gathered in the Italian Research Infrastructures covering the environmental scientific domains is very ambitious considering, at present, the vast heterogeneity and the different developmental stages of the Research Infrastructures. Therefore, particular attention and efforts are required in harmonization integration and standardization processes starting from metadata creation. In this context, a fundamental aspect regards the application of the FAIR principles of Findability, Accessibility, Interoperability, and Reusability not only to the data produced in the platforms but also to the whole scientific process that leads to the data. The FAIR data principles, defined in 2016 by a consortium of scientists and organizations, describe distinct considerations for contemporary data publishing environments with respect to supporting both manual and automated deposition, exploration, sharing, and reuse. In detail, according to the FAIR guiding principles, data need to be (1) Findable (e.g., data are described with rich metadata), (2) Accessible (e.g., data and metadata are retrievable by their identifier using a standardized communications protocol, and the protocol is open, free, and universally implementable), (3) Interoperable (e.g., data and metadata use a formal, accessible, shared, and broadly applicable language for knowledge representation), and (4) Reusable (e.g., data and metadata are released with a clear and accessible data usage license). Note that these high-level FAIR Guiding Principles precede implementation choices, and do not suggest any specific technology, standard, or implementation-solution. However, these principles can be used as a guide for the design and development of the proposed platform architecture. With this regard, quality and rich metadata repositories are essential for managing large numbers of datasets and conducting appropriate data selection, enhancing accessibility, interoperability and data reuse, and revealing the fundamental importance of quality metadata in data management

During the first year, new personnel were hired and the digitalization of the three ANAEE platforms was discussed. A format of a meta database was developed as the first entailed action for conducting harmonization and standardization processes of the ITINERIS Research Infrastructure and developing the ITINERIS Hub. Acquisition, and installation of the scientific instrumentation and technological equipment at each site/platform, as well as data collection will be completed in the second year. The third year will be spent implementing the platform set-up, full operation and transfer of data.

2. FREE-AIR MANIPULATIVE TERRESTRIAL ECOSYSTEM PLATFORMS ANAEE-ITALY TO EVALUATE CLIMATE CHANGE DRIVER EFFECTS

2.1 The effects of Nitrogen deposition on forest ecosystems - Ndep

Human activity has critically altered the nitrogen cycle during the last decades, thus resulting in increased nitrogen deposition in all ecosystems. The production of fertilizer and fossil fuel combustions have increased the amount of N reactive compounds such as NO_X and NH_3 to double compared to preindustrial levels, with unclear short- and long-term effects on forest ecosystems. In non-limited environments, nitrogen addition can induce growth reduction, increase plant sensitivity to other stress and/or disturbances (i.e., pests, drought), and lead to ecosystem nitrogen saturation, with severe consequences such as a general decrease in forest productivity and biodiversity.

To highlight the existing uncertainties related to the potential long-term nitrogen cumulative effects on forests exposed to elevated atmospheric nitrogen loads, accounting for the slow processes involved in forest ecosystems, in 2015 the Italian Nitrogen Deposition Network (INDN) developed an ANAEE Research platform named Ndep. The platform consists of long-term manipulative experiments in different forest sites and climatic conditions, that simulate elevated nitrogen deposition where the role of canopies as layers for nitrogen interception and absorption are evaluated for more realistic future scenarios about the long-term nitrogen deposition effects on forest ecosystems.

Ndep sites information:

- Site Cansiglio (46° 3′ 19” N 12° 22′ 51″ E at 1100 m a.s.l.), contact Prof. Rossella Guerrieri (Università degli Studi di Bologna, email - r.guerrieri@unibo.it) and Prof. Federico Magnani (Università degli Studi di Bologna, email - federico.magnani@unibo.it) ;
- Site Cembra (46°12'9"N, 11°12'35"E, at 1270 m a.s.l.), contact Dr. Damiano Giannelle (Fondazione Edmund Mach, email - damiano.gianelle@fmach.it);
- Site Collelongo (41° 50′ 59” N 13° 35′ 6″ E at 1560 m a.s.l), contact Dr. Francesco Mazzenga (CNR-IBE, email - francesco.mazzenga@cnr.it), Dr. Alessandro Messeri (CNR-IBE, email - email: alessandro.messeri@cnr.it);
- Site Monticolo (46°255’35″ N; 11◦17́55″ E, 550 m a.s.l.), contact Dr. Maurizio Ventura (Università degli Studi di Bolzano, email - maurizio.ventura@unibz.it).

2.2 The effects of Carbon dioxide on crop production - Durum FACE

The rising CO_2 concentrations in the atmosphere caused by fossil fuel combustion are considered the major drivers of rising temperatures and climate change. The higher CO_2 concentrations directly influence plant growth and physiology improving their photosynthetic activity and reducing their stomatal conductivity, thus enhancing water use efficiency, which helps to overcome drought conditions. As a result, elevated CO_2 concentrations enhance above and below-ground biomass, as well as harvestable yield of crops. Despite the beneficial effect on crop biomass, several studies highlighted the uncertainties of elevated CO_2 concentration on crop nutritional properties. Further, the relationship between elevated CO_2 and other climate change effects such as increased temperatures, and extreme events such as heat waves, drought and changes in precipitation patterns cause relevant concerns related to agricultural crop production.

To evaluate the effects of rising CO_2 on plant growth, the Research Infrastructure Durum FACE (Free Air CO_2 Enrichment) will be set up during the spring of 2025. This facility will investigate the crop’s response to elevated CO_2 in concomitance with other stressors (i.e., drought) to untangle concerns related to agricultural crop production, food security and food safety under climate change scenarios.

Durum FACE - Free Air CO2 Enrichment site information:

- Fiorenzuola d'Arda, (PC), contact Dr. Luigi Cattivelli (CREA, email - luigi.cattivelli@crea.gov.it), Dr. Davide Guerra (CREA, email - davide.guerra@crea.gov.it), Dr. Alessandro Zaldei (CNR-IBE, email - alessandro.zaldei@cnr.it).

2.3 The effects of Tropospheric ozone on plants - FO3X

Tropospheric ozone (O_3) is an air pollutant and one of the major greenhouse gases. This highly reactive and oxidative gas occurs naturally in the environment; however, it has increased in the troposphere since the industrial revolution, resulting from photochemical reactions of its precursors such as methane, carbon monoxide, volatile organic compounds and nitrogen oxides. O_3 negatively affects plant biochemistry and physiology, damaging cells and tissues causing dysfunctions in stomatal control, altering photosynthesis and carbon assimilation, producing signs of foliar injury and accelerating leaf senescence, leading to impairment of plant functioning. Although a reduction in tropospheric O_3 peaks was obtained in some areas of the globe, the high O_3 concentrations remain relevant global concerns for plant functioning and ecosystem services such as crop production in response to O_3.

To evaluate the effects of O_3 on plants alone and in concomitance with other stressors, such as soil water stress, nutrient alterations, salinity etc. the FO3X (Ozone FACE - Free air-controlled exposure) Research Infrastructure exposes plants continuously day and night at three levels of O_3. The FO3X is the only facility simulating the effects of elevated concentrations of O_3 on plants in concomitance with other stressors in a Mediterranean Climate, thus representing a fundamental facility to develop appropriate plant risk assessment against future global changes.

FO3X - Ozone FACE - Free air-controlled exposure site information:

- Via Madonna del piano 10, 50019, Sesto Fiorentino (FI), contact Dr. Yasutomo Hoshika (CNR-IRET, email - yasutomo.hoshika@cnr.it)

2.4 The data collected in the platforms

2.4.1 Biological, physical, and chemical data from manipulated ecosystem experiments

The harmonized database is designed to collect data generated from free-air manipulative experiments conducted at the Ndep, Durum FACE, and FO3X platforms. These include detailed measurements of biological, physical, and chemical attributes of ecosystems, such as plant growth and phenology, photosynthetic parameters, tissue nutrient content, soil composition and moisture, and atmospheric gas concentrations. Such data are critical for assessing the impacts of increased nitrogen deposition (Ndep), elevated CO_2 levels (Durum FACE), and exposure to tropospheric ozone (FO3X) on vegetation.

2.4.2 Near real-time meteorological and environmental data

The platforms are equipped with advanced sensors to support automated acquisition of meteorological and environmental data, including temperature, humidity, precipitation, solar radiation, and wind speed and direction. In particular, the FO3X platform provides near real-time data on ozone concentrations and weather conditions, which are accessible in both JSON and CSV formats. Data can be collected at varying temporal resolutions (e.g., hourly averages or minute-level data), ensuring flexible and efficient data flow management tailored to research needs.

2.4.3 Structured metadata and documentation of experimental activities

In addition to experimental data, the database hosts rich and structured metadata describing the experimental design, site characteristics, measurement techniques, and instrumentation used. These metadata, developed in compliance with the FAIR principles, also include information on data ownership, licensing, units of measurement, and acquisition protocols. This comprehensive documentation ensures traceability, interoperability, and long-term reusability of the data, enabling seamless integration across platforms and centralized access through the ITINERIS Hub.

3. PLATFORM IMPLEMENTATION WITH INSTRUMENTS AND TECHNOLOGIES TO IMPROVE HARMONIZATION AND STANDARDIZATION PROCESSES FOR PROJECT HUB CREATION

The responsible in charge of each site belonging to open-air terrestrial platforms was primarily contacted, and several meetings were carried out with the various team members of each research group to have a comprehensive understanding of all research activities and data collection taking place at each site. Currently, the platforms at each site collect the data independently, and there is no uniformity nor homogeneity of the data type and format. Several meetings were carried out to discuss some of the fundamental aspects relevant to WP6.3.

Therefore, we discussed with each group team the matter related to data acquisition and defined shared strategies to facilitate the process of data integration and harmonization as well as data availability over time. We expect to establish specific agreements and adopt shared policies regarding the acquisition of past and future data collected, taking into consideration the specific needs and complexity of each site and experimental manipulation. ITINERIS project aimed at purchasing scientific instrumentation and technological equipment to improve research activities and automatization in data collection at each of the ANAEE sites. Specific instrumentations are necessary to respond to the manipulative experiment peculiarities and the plant species subjected to the experiments. At present, we have completed the purchase of several instruments whose temporary installation at each platform is taking place. A number of them remain in the purchasing process.

The implementation of the research platforms at each site with new instrumentations and similar policies regarding data acquisition and sharing on dedicated informatics infrastructures are prerequisites to guarantee the accomplishment of WP6.3.

3.1 Development of the database for the Ndep, Durum FACE and FO_3X platforms

The creation of high-quality and comprehensive metadata is a key step to enable efficient exploration and informed selection of datasets within the ITINERIS Hub, which is expected to host a wide array of data collections. To support this, we have developed a standardized metadatabase format derived from the Ndep network. This format adheres to the FAIR principles and captures all relevant qualitative and quantitative information associated with the datasets. It includes detailed descriptions of manipulative experiments, site conditions, monitoring techniques, sensor specifications, and the methodologies used to assess biological, physical, and chemical attributes. This metadatabase serves as a foundational tool to harmonize and integrate data across the Ndep, Durum FACE, and Fo3X platforms.

A meeting was held with the team members of the ITINERIS Work Package 2 (WP 2), whose objective is to establish a unified framework for facilitating access to facilities, data, and associated services of the ITINERIS Hub. The collaboration with the WP 2 team was instrumental in advancing and refining the adoption of FAIR (Findable, Accessible, Interoperable, and Reusable) principles within the meta database. This effort aims to enhance data interoperability across platforms.

Guided by the WP 2 team, key elements of the meta database—such as semantics, syntax, and terminology—were thoroughly assessed and are currently undergoing active discussion and iterative refinement. This ongoing process is expected to lead to a more cohesive approach to data management and accessibility, improving data access to stakeholders and users of the ITINERIS repositories Hub. The meta database is structured according to the RESTful APIs (Application

Programming Interface) specification. The RESTful API, also known as REST (Representational State Transfer), is an application programming interface (API or web API) that conforms to the constraints of REST architectural style and allows for interaction with RESTful web services. When a client (e.g., researcher) request is made via a RESTful API, it transfers information on the state of the resource to the requester or endpoint. This information, or data, is delivered in several formats via HTTP such as JSON (Javascript Object Notation), HTML (HyperText Markup Language), XML (eXtensible Markup Language), or plain text as CSV (comma-separated values). JSON is the most generally popular file format to use because, despite its name, it's language-agnostic, as well as readable by both humans and machines. For this reason, the principal data format of metadata is based on JSON. The use of JSON guarantees interoperability across systems since it is manageable by all programming languages such as Python, R, Java and so on. In addition, other benefits of using the JSON format for the metadata can be attributed to its readability (JSON is easy to read as it's based on defining objects and values), parsing speed in the workflows, and small size.

The meta database implementation is based on the Model-View-Controller (MVC) architectural model, which was implemented through Spring boot, an open-source Java framework. The Model-View-Controller (MVC) pattern is an architectural design pattern that separates an application into three interconnected components:

The Model represents the application's data and business logic (Figure 1). It is responsible for managing the data and the rules that govern it. In the implementation, the Model is represented by the Service Layer and the Repository Layer. The Service Layer contains the core business logic, operations, and algorithms of the application. The Repository Layer, also known as Adapter or Persistence Layer, is responsible for data base operation, and thus abstracts CRUD (Create, Read, Update, and Delete) operations. To ensure data permanence, MongoDB was adopted as the solution. MongoDB is a NoSQL (i.e., non-relational) database which uses a flexible JSON-like document model to store data in collections. As a result of this solution, the system has been able to manage data more dynamically and scalably since data can be added and modified without first defining a schema.

The View represents the presentation layer and is responsible for displaying the data to the user (Figure 2). It handles user interface elements and rendering. In the meta database the view is implemented using Swagger UI. Swagger UI is an open-source tool that helps to visualize and interact with the REST APIs and helps to explore and test RESTful APIs created, by providing a user-friendly interface for developers or consumers to browse API documentation, test API endpoints, and observe with different parameters and options. Alternatively, more experienced users can query the information via endpoints in major programming languages such as Python, R, and Java, integrating this information into their scientific analyses workflow.

The Controller acts as a middle layer between the Model and View. It receives user input, processes it, and manages the flow of data between the Model and View. In Spring boot, the controllers handle user requests, manage the interaction between the Model and View, and control the application's flow. In addition, endpoints are exposed through the controller according to the RESTful APIs (Application Programming Interface) specification. By using the exposed endpoints, a user can request and receive JSON-formatted information directly from the system.

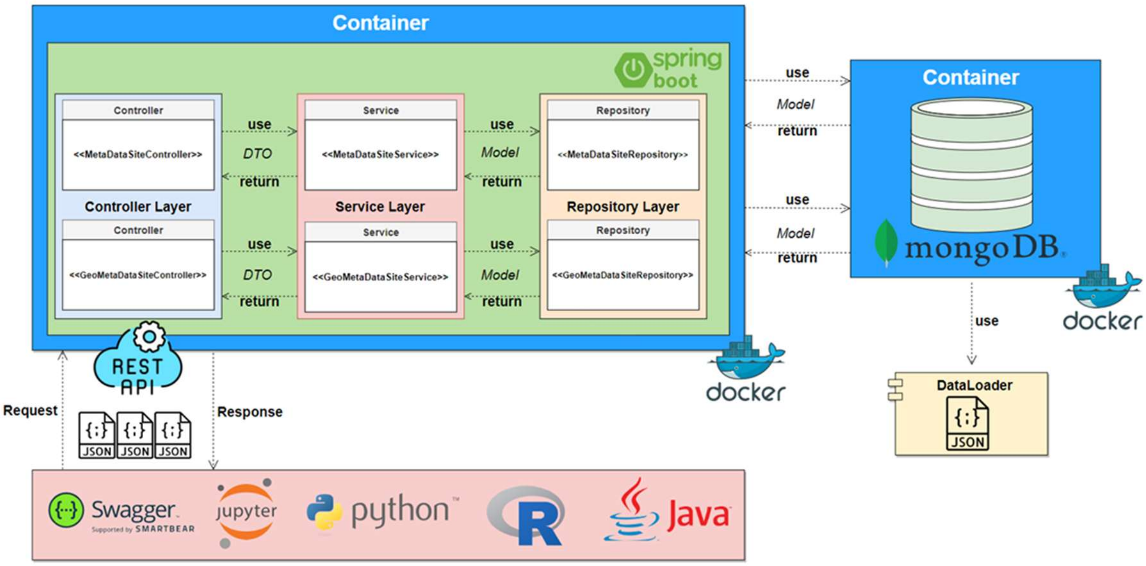


Figure 1 Graphical representation of the Model-View-Controller architectural design adopted for the ITINERIS meta database.

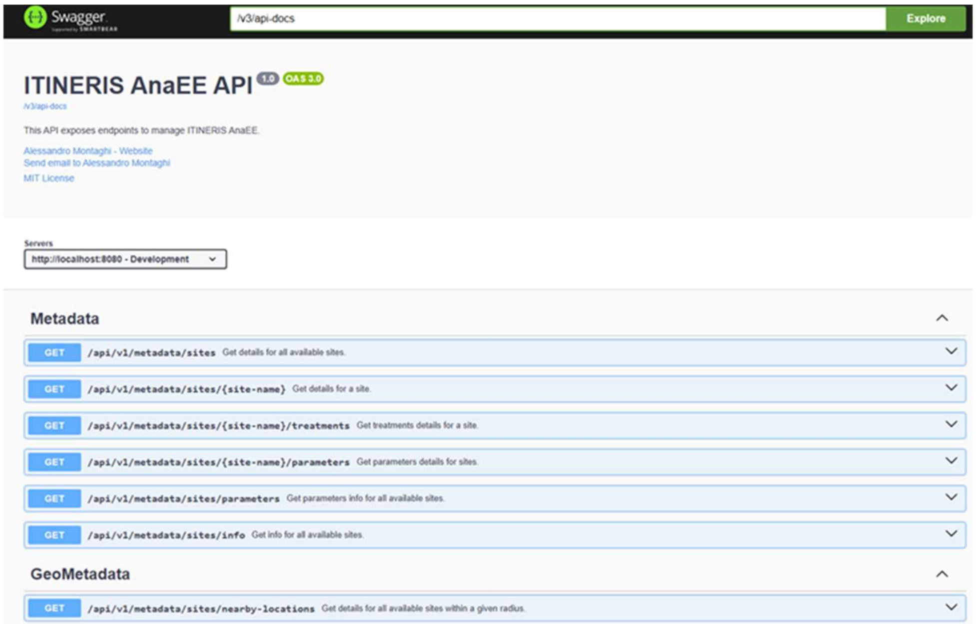


Figure 2 Presentation layer implemented with Swagger UI for metadata retrieval, update and download.

3.1.1 REST API Documentation Tool

In the real-world scenarios of software development, APIs (Application Programming Interfaces) are the backbone of communication between systems and applications. Clear, interactive, and accessible documentation is essential to ensure developers understand how to use APIs correctly, minimizing errors and inefficiencies. Swagger UI, the solution adopted in our application, stands out as one of the most popular and reliable tools for documenting REST APIs. This text explores the key reasons why Swagger UI should be your go-to tool for API documentation.

In the software application described above, the swapper UI is chosen for the following reasons.

1. **Interactive User Interface**: One of Swagger UI's greatest strengths is its highly intuitive and interactive user interface. Unlike traditional static documentation that often requires developers to interpret text and manually build API requests, Swagger UI allows users to explore and test APIs directly. Developers can send requests, configure parameters, and observe responses straight from the interface, significantly reducing the time needed to understand an endpoint's functionality. For instance, an endpoint like *"GET/users"* is displayed with details about available parameters, required headers, and the expected response format. Users can easily input desired data into provided fields and click "Execute" to view real-time results. This functionality eliminates the need for additional manual steps, such as using separate tools for API testing.

2. **Standardization with OpenAPI**: Swagger UI is built on the OpenAPI standard (formerly known as Swagger Specification), a widely adopted specification for describing REST APIs. By using Swagger UI, organizations ensure uniform and structured documentation across all their APIs. The OpenAPI standard provides a clear model to describe endpoints, parameters, requests, responses, and errors, making documentation readable by both humans and machines. For example, an OpenAPI definition for an endpoint specifies not only the HTTP method and URL but also details like data types, default values, and validation constraints. Swagger UI automatically translates this definition into an elegant user interface, ensuring developers have accurate, up-to-date information.

3. **Enhancing Team Collaboration**: In complex development environments where multiple teams work on interdependent projects, Swagger UI promotes transparent communication and shared understanding of APIs. Designers can define APIs in an OpenAPI file, and developers can explore and test them directly through Swagger UI, eliminating ambiguity and misunderstandings. Furthermore, Swagger UI's features make it easier for QA teams to test APIs without needing to write code or configure complex tools. Even non-technical stakeholders, such as product managers, can use Swagger UI to understand API functionalities, enabling more informed decision-making.

4. **Easy Integration and Deployment**: Swagger UI is easy to integrate into any project. It is available as a JavaScript library that can be embedded in web applications or deployed as a standalone application. Additionally, it supports both JSON and YAML formats for OpenAPI specifications, allowing organizations to choose the format that best fits their needs. Deploying Swagger UI on a server enables all developers to access the API documentation via a web browser. This approach eliminates the need to distribute static documents and ensures the documentation is always in sync with the actual API implementation.

5. **Support for Secured APIs**: Many modern APIs require authentication and authorization to access. Swagger UI provides native support for various authentication methods, including OAuth 2.0, JSON Web Tokens (JWT), API keys, and Basic Authentication. Users can easily configure access tokens or other credentials directly in the interface, making API testing seamless.

6. **Improved Developer Experience**: With its well-designed interface and user-friendly features, Swagger UI significantly enhances the developer experience. The ability to expand and collapse endpoints, search for specific functionalities, and view example payloads for requests and responses makes documentation accessible even to those unfamiliar with the API. Another advantage is the clear display of potential error responses, helping developers anticipate and handle edge cases effectively during API implementation. This feature is particularly useful during the development of complex or multi-user applications where secure API access management is critical.

7. **Time-Saving and Error Reduction**: Manually documenting APIs is time-consuming and error-prone, especially for projects that evolve constantly. Swagger UI automates much of the documentation process, eliminating discrepancies between the documentation and the actual API implementation. Any changes to the OpenAPI specification are automatically reflected in Swagger UI, ensuring developers always have access to the most recent and accurate documentation.

3.1.2 Data allocation and security

To achieve the goal of integrating and harmonizing the Italian free-air manipulative platform data of ANAEE terrestrial ecosystems, the platform should meet the principles of Findability, Accessibility, Interoperability, and Reusability (FAIR).

In this regard, most of the solutions for data science platforms are based on producing ETL (extract, transform, load) or ELT (extract, load, transform) processes or custom hybrid solutions. About ETL solutions, the first step in any ETL (and ELT) scenario is data extraction. The ETL extraction step is responsible for extracting data from the source systems. The *Extract* component needs to be designed to have the ability to extract from different data sources, such as from structured data sources (e.g., relational databases, data sensors), unstructured data sources, semi-structured data sources (e.g., XML and JSON generator), legacy systems, application packages, etc. The processes for data collection can be scheduled according to several different policies to cope with static data (changing sporadically), quasi real time data (changing a few times a day) to real time data (changing every few minutes, such as the weather data) and considering all the permission accesses connected to each different piece of information managed in the platform.

The second step in any ETL scenario is data *Transformation*. The task of transformation component is data cleaning, transformation, integration, and harmonization. The transformation component can be separated into sub-components that each focus on a single task. In this way, it can be easily scaled, extended, and maintained. For example, through a subcomponent called Metadata Manager, metadata can be generated automatically.

Finally, in the *Loading* step, extracted and transformed data is written into the dimensional structures actually accessed by the end users and application systems. Loading step includes loading data in a database. Based on the type of data, different types of databases can be chosen. For structured data type the most common database implementation is based on the relational model which uses SQL as its query language. However, NoSQL (Not Only SQL) database solutions are becoming more prominent when data are typically non-structured, complex and does not fit well into the relational model.

3.1.3 Training with Jupyter Notebook

Training users to effectively utilize an API portal connected to a metadata-based system powered by MongoDB requires a platform that is accessible, interactive, and adaptable to various learning styles. Jupyter Notebook stands out as an ideal choice for implementing such training. By combining live code execution, explanatory text, and visualizations, Jupyter enables users to explore complex concepts and workflows in a hands-on, intuitive way. Here's why Jupyter Notebook is a perfect fit for this purpose.

1. **Interactive and Practical Learning**: Jupyter Notebook allows users to interact directly with the API and MongoDB system, providing immediate feedback and fostering a practical understanding of how the system operates. Instead of passively watching demonstrations or reading documentation, users can engage with the system in real time.

    1.1 API Exploration: Users can execute requests to the API and observe the responses instantly, learning how to retrieve and manipulate metadata.

    1.2 Database Interaction: users can run MongoDB queries within the same notebook, seeing how their actions via the API affect the underlying database.

2. **Combines Explanations with Real-World Demonstrations**: Jupyter Notebook's ability to mix explanatory text, live code, and visualizations ensures users not only learn what to

do but also understand why it's done. This combination is essential for explaining abstract concepts, such as metadata structures and query logic.

2.1 **Theory with Context**: Trainers can include Markdown cells to explain MongoDB's schema design for storing metadata or the purpose of specific API endpoints.

2.2 **Live Examples**: Code cells following the explanations let users immediately test their understanding by executing practical examples.

3. **Real-Time Experimentation and Immediate Feedback**: Users often learn best through experimentation, and Jupyter Notebook facilitates this by allowing them to try out different API calls, database queries, and parameter configurations in a controlled environment. Immediate feedback helps users understand what works, what doesn't, and why. For example, users can test various parameters for retrieving or updating metadata and instantly see the results (API Requests) and users can explore the underlying database to see how data is stored or aggregated (MongoDB Queries).

4. **Structured Problem Solving and Scenarios:** Jupyter Notebook enables trainers to design structured workflows that simulate common tasks or challenges users might encounter while working with the API and database. These workflows can include Guided Tasks (i.e., Step-by-step instructions to query specific metadata or update entries) and Scenario-Based Exercises (i.e., Challenges like finding metadata anomalies or generating summaries using MongoDB's aggregation framework).

5. **Error Handling and Troubleshooting**: a key aspect of training is preparing users to handle errors or unexpected behavior confidently. Jupyter Notebook provides an environment where users can safely encounter and resolve common issues, such as non-correct API Requests (e.g., demonstrating how to identify and fix incorrect parameters).

3.2 Manual for the management of the database

This section describes the structure and management of the harmonized database designed to support data integration across the three experimental platforms: Ndep, Durum FACE, and FO3X. The database has been developed to accommodate the diverse types of data generated by these platforms, ensuring consistency, traceability, and alignment with FAIR principles.

In the following sections, an example configuration is provided based on the FO3X platform. This serves to illustrate the database structure and metadata requirements. The database is fully prepared to ingest and organize incoming data from all three platforms, supporting future data entry, exploration, and analysis in a unified and coherent format.

A short description of the platform can be found in the first page of the database

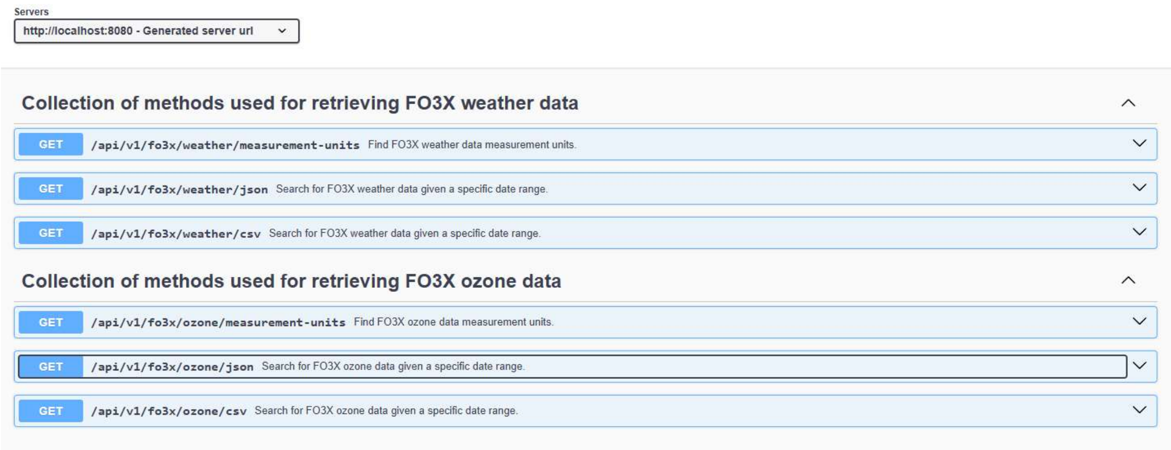
Swagger is a very useful tool for documenting and testing REST APIs created with Spring Boot. It allows you to generate interactive API documentation in a readable, browser-accessible format. A Swagger component was implemented for the client side. Swagger is a very useful tool for documenting and testing REST APIs created with Spring Boot. Specifically, it allows interactive API documentation to be generated in a readable, browser-accessible format. Through Swagger it was possible to:

1. Describe and document APIs that distribute (provide) data, for example from a database.
2. To expose data in a structured and accessible way to client.

For completeness, Swagger does not distribute the data itself, but serves to:

1. Clearly describe how to access that data via REST APIs.
2. Allow other systems or developers to easily discover and test how to retrieve the data.

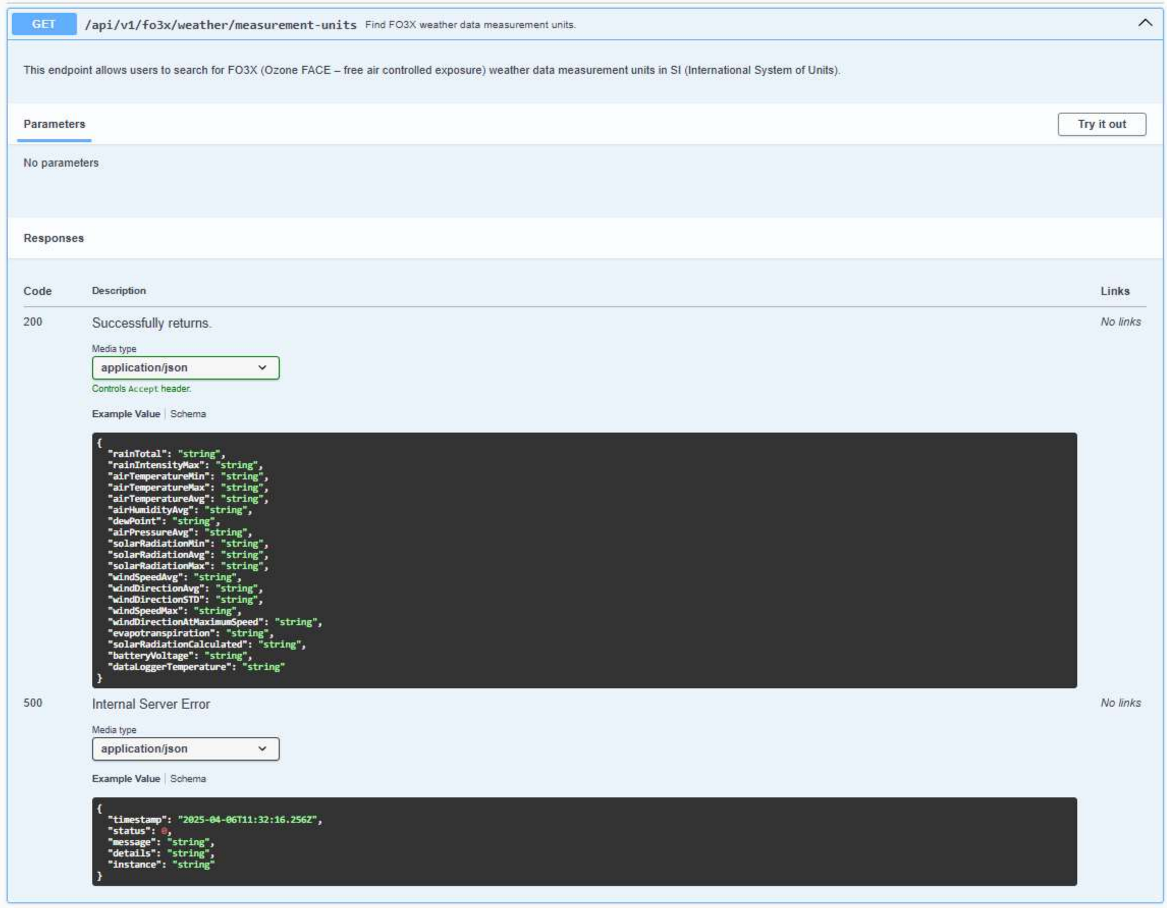
3. Standardize how the data is exchanged (JSON payload, param queries, etc.).



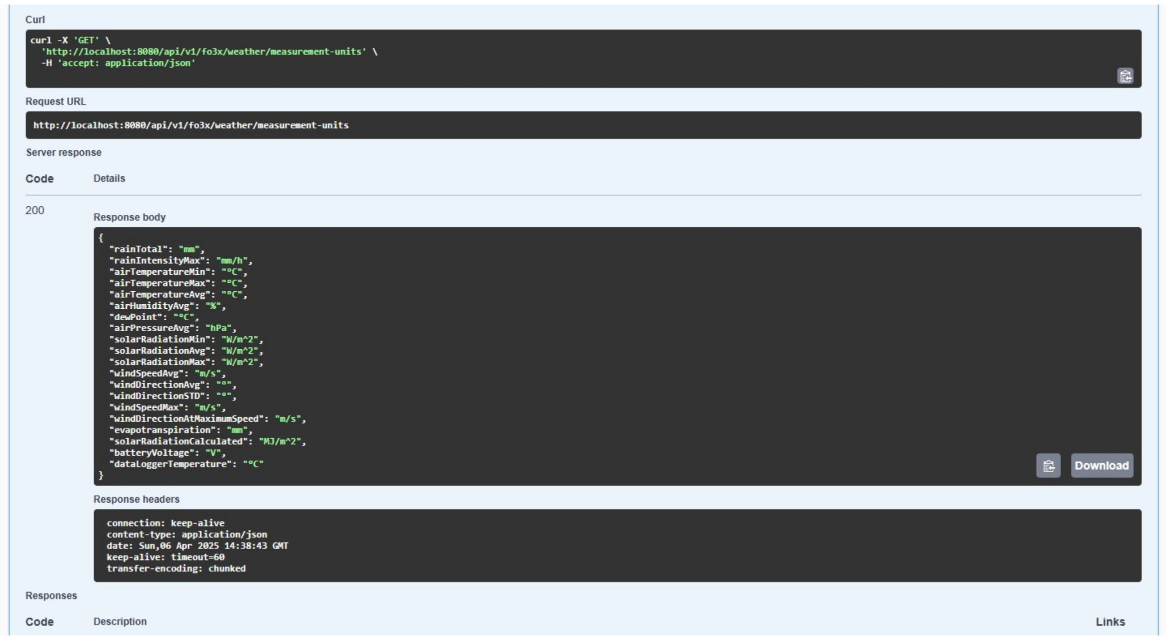
In designing the services in Swagger, it was decided to split into 2 collections (Collection of methods used for retrieving FO3X weather data and Collection of methods used for retrieving FO3X ozone data), each consisting of 3 GET-type endpoints, to separately request data from the FO3X weather and ozone sensors. The three endpoints implemented for each collection are as follows:

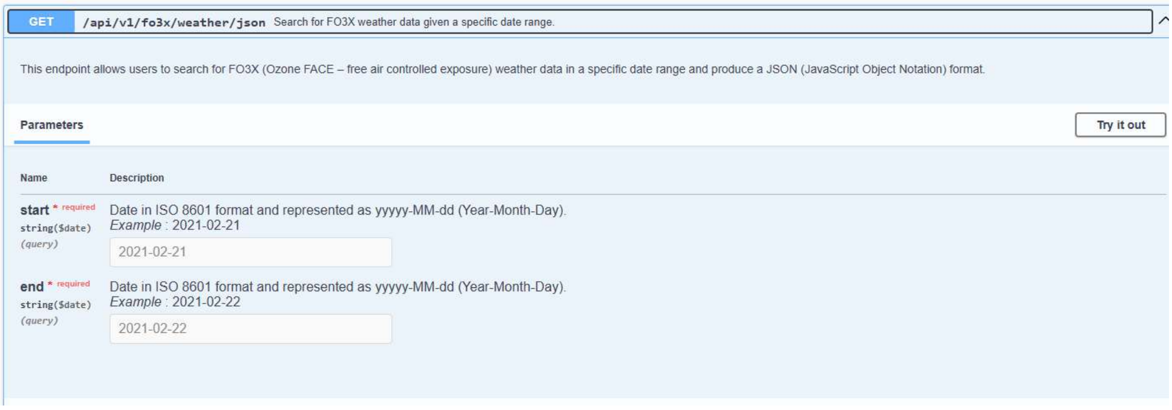
1) Collection of methods used for retrieving FO3X weather data.
   1.1 **measurement-units**, to find FO3X weather data measurement units.
   1.2 **json**, to retrieve FO3X weather data in json format (JavaScript Object Notation), given a specific data range.
   1.3 **csv**, to retrieve FO3X weather data in csv (Comma-separated values) format, given a specific data range.

2) Collection of methods used for retrieving FO3X ozone data.
   2.1 **measurement-units**, to find FO3X ozone data measurement units.
   2.2 **json**, to retrieve FO3X weather data in json format (JavaScript Object Notation), given a specific data range.
   2.3 **csv**, to retrieve FO3X weather data in csv (Comma-separated values) format, given a specific data range.

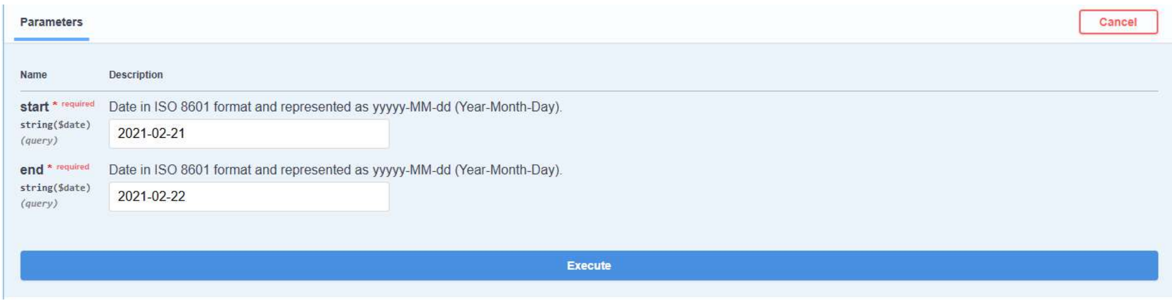


The first endpoint of the FO3X (Ozone FACE - free air-controlled exposure) weather data collection is used to obtain the units of the values obtained from the remaining endpoints. Below is a screenshot of the output of the requested data in JSON format.

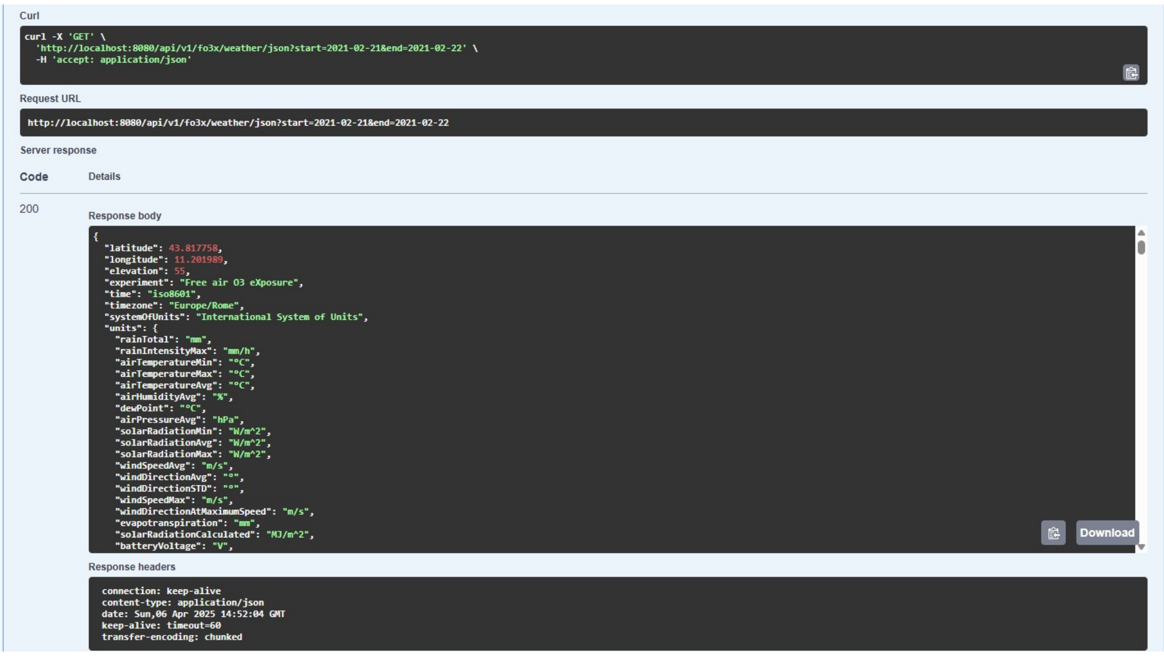




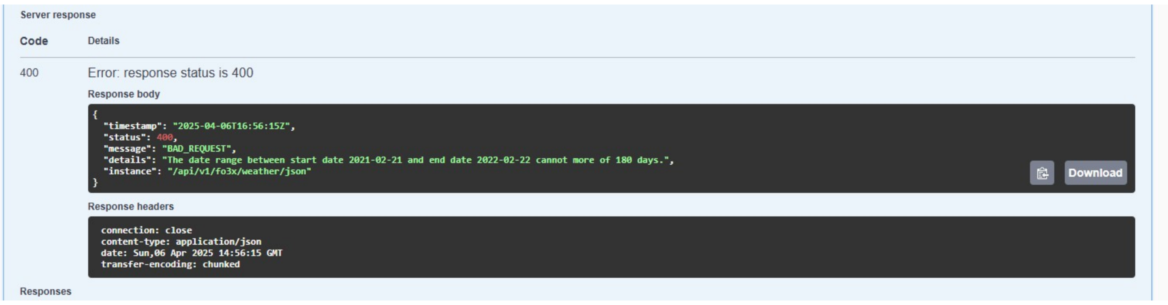
The second endpoint of FO3X weather data collection allows users to search FO3X weather data (Ozone FACE - free air-controlled exposure) in a specific date range and produce a JSON (JavaScript Object Notation) format. There are two input parameters, namely start date and end date, each accepting a date in ISO 8601 yyyy-MM-dd (Year-Month-Day) format.



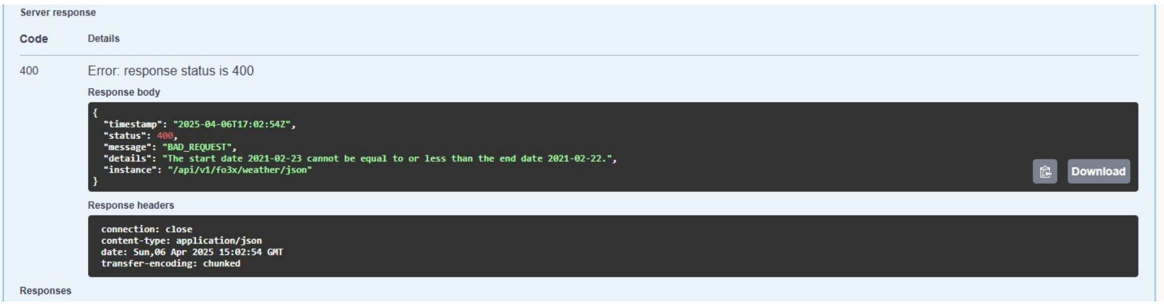
If both dates are entered correctly, a JSON file is returned, which can be downloaded at will, containing the time data of the variables contained in the database



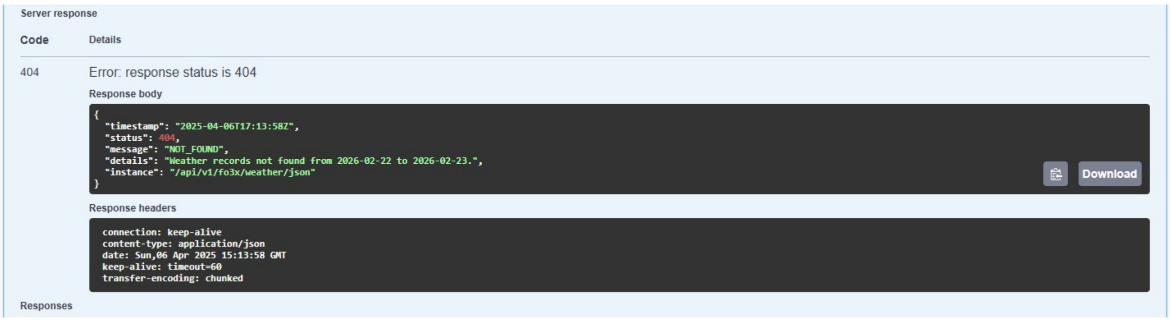
In case there is an error on the dates entered you get error messages warning the user. For example, in case there is an error on the dates entered you get error messages warning the user. For example, if the interval between the start and end dates exceeded 180 days, the following error is obtained.



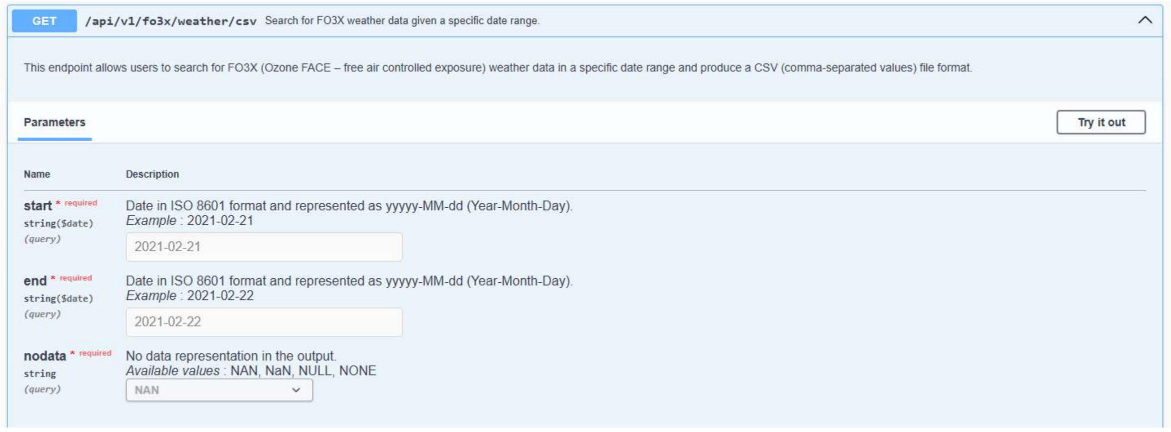
When the start date is later than the end date, the following error is generated:



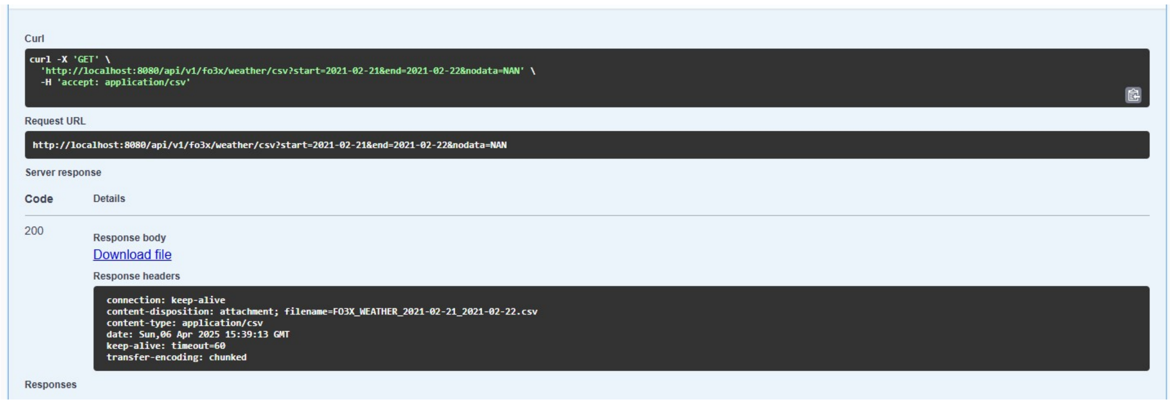
Finally, if the date range contains no data value, an exception is raised



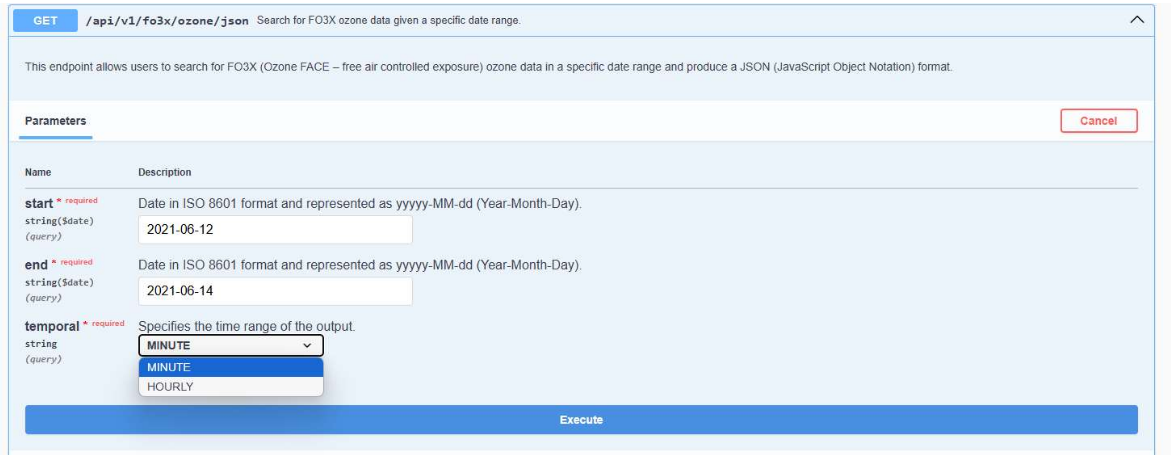
The last endpoint allows users to search for FO3X (Ozone FACE – free air controlled exposure) weather data in a specific date range and produce a CSV (comma-separated values) file format. Also for this endpoint, the user enters two input dates, and if no error occurs, a button is activated where the user can download the data as a CSV file. In addition, the user can decide how to represent empty values, via NAN, NaN, NULL, NONE

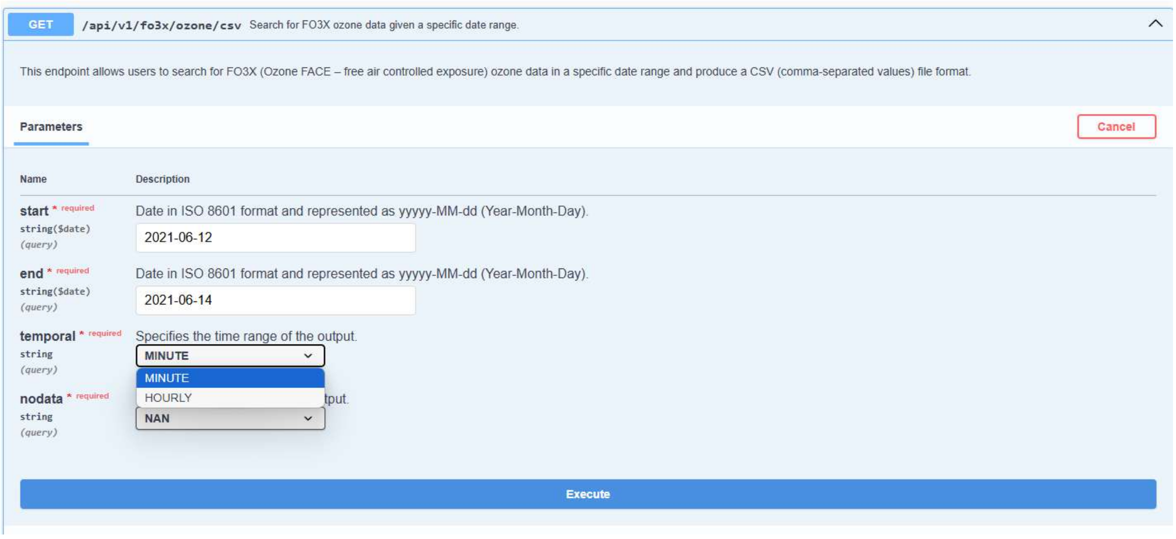


If the parameters are correct, it returns a link from where to download the CSV file:



Regarding the collection of methods used to retrieve FO3X ozone data, we have that the only difference turns out to be that the data can be obtained with time step of the minute or as an hourly average.





However, to better manage the transmission of data in the network, the time range for minute data has been set at 3 days, while that for hourly data is remitted to 180 days.

4. CONCLUSIONS

The database described in this manual is fully operational and ready to receive incoming data from the FO3X, Durum FACE, and Ndep platforms. In alignment with the goal of completing the repositories for automatization, near real-time transfer, and long-term storage of data, the system has been designed to support continuous data updates as they are collected by the platforms. Thanks to its flexible architecture and adherence to FAIR principles, the database ensures that all harmonized data can be seamlessly transferred to the central ITINERIS Hub. As such, this infrastructure represents a key milestone in the completion of the open-air terrestrial platform repositories, enabling efficient data management, accessibility, and long-term exploitation of the scientific outputs generated across the network.