PROCEEDINGS OF SCIENCE

# Open-source and cloud-native solutions for managing and analyzing heterogeneous and sensitive clinical Data

**Daniele Spiga,**[a,*] **Diego Ciangottini,**[a] **Alessandro Costantini,**[c] **Sara Cutini,**[a] **Cristina Duma,**[c] **Jacopo Gasparetto,**[c] **Pasquale Lubrano,**[a] **Barbara Martelli,**[c] **Elisabetta Ronchieri,**[c] **Davide Salomoni,**[c] **Giusy Sergi,**[c] **Loriano Storchi**[b] **and Mirco Tracolli**[a]

[a] *Istituto Nazionale di Fisica Nucleare, Sezione di Perugia, Via A. Pascoli, 06123 Perugia, Italy*

[b] *Dipartimento di Farmacia, Universitá degli Studi G. D'Annunzio, 66100 Chieti, Italy*

[c] *Istituto Nazionale di Fisica Nucleare, CNAF, Viale Berti Pichat 6/2, 40127 Bologna, Italy*

*E-mail:* daniele.spiga@pg.infn.it

The requirement for an effective handling and management of heterogeneous and possibly confidential data continuously increases within multiple scientific domains. PLANET (Pollution Lake ANalysis for Effective Therapy) is a INFN-funded research initiative aiming to implement an observational study to assess a possible statistical association between environmental pollution and Covid19 infection, symptoms and course. PLANET is built on a "data-centric" based approach that takes into account clinical components, environmental and pollution conditions, complementing primary data and many eventual confounding factors such as population density, commuter density, socio-economic metrics and more. Besides the scientific one, the main technical challenge of the project is about collecting, indexing, storing and managing many types of datasets while guaranteeing FAIRness as well as adherence to the prescribed regulatory frameworks, such as those granted by the General Data Protection Regulation, GDPR. In this contribution we describe the developed open-source DataLake platform, detailing its key features: the event-based storage system provided by MinIO, which allows automatic metadata processing; the data-ingestion pipeline implemented via Argo Workflows; the GraphQL interface to query object metadata; finally, the seamless integration of the platform within a compute multi-user environment, showing how all these frameworks are integrated in the Enhanced PrIvacy and Compliance (EPIC) Cloud partition of the INFN Cloud federation.

*International Symposium on Grids & Clouds 2022 (ISGC 2022)*
*21 - 25 March, 2022*
*Online, Academia Sinica Computing Centre (ASGC), Taipei, Taiwan****

*Speaker





PoS(ISGC2022)022

## 1. Introduction

Any data with an high variability of either data types or format can be considered "heterogeneous" and they are often acquired by distinct sources and combined together to be analyzed. This lead to treating them with a scope which is completely different with respect the original one. The presented integration is finalized to meet the demand specifically coming from a data analysis over such heterogeneous datasets.

An important point when treating this kind of data is that they can differ not only in term of data type or format definition, but, even worse, homogeneity still cannot be guaranteed within a given data type. As an example, when looking at historical data, information coming within a certain time frame can differ from another one depending on the methodologies adopted for its manipulation.

From a data analyst perspective there is no control on the data acquisition workflow and thus on the procedure, if any, used for data model definition. This is completely in contrast with a classical scientific experiment where the data model definition is one of the first steps to be implemented.

For all these reasons heterogeneous data are possibly ambiguous and redundant and, moreover, they are very often of uncertain quality. For instance there is no guarantee not to have missing values in any dataset. Also the lack of data description poses a challenge for what concern data archival and data classification, i.e. the so-called "findability".

In addition, when heterogeneous data become also sensitive (for Medical Physics or other fields) the level of complexity increases, in fact data handling and processing requires system certification, i.e. both organizational and technical security measures must be adopted to hosts those type of data and, then, to support the related processing and analysis.

INFN is a pioneer institution in the design and implementation of large-scale computing infrastructures and applications, primarily developed to meet the needs of the latest generations of high energy physics (HEP) experiments, now rapidly extending to other communities also in the context of multidisciplinary research projects. The values that INFN can bring in this challenge is summarized in three main areas: Data Analysis, to develop models and to implement statistical data analysis; Data Curation and Data Management to organize and to integrate data, even collected from heterogeneous sources; Integrated and Certified Computing Infrastructure provisioning, to enable both data archival and data processing. The latter includes A ISO/IEC 27001, 27017, 27018 Certified infrastructure. This paper describes several software solutions developed in the context of the PLANET project and coherently integrated following a cloud-native approach reflecting the vision of INFN Cloud [1] and the related security technical and organizational measures, guidelines and processes developed within the EPIC-Cloud [2]. In sec 2 we introduce the PLANET project and the challenges it poses. Sec 3 we provide a comprehensive description of the requirement analysis as well as the developed solution. Sec 4 an overview of the final architecture is reported, while in Sec 5 we show the strategy toward a proper handling of sensitive data. Finally a summary and future steps are summarized in Sec 6.

## 2. The PLANET use case

PLANET is an INFN-funded research initiative developed in synergy with the epidemiological and medical knowledge at the University of Perugia. It aims at developing an observational

(ecological) study to evaluate the association between air pollution and Covid19, being aware that severity Covid19 disease and deaths are also influenced by many variables (age, gender, comorbidities, frailty, etc..). The study is based on the hypothesis that pollution may contribute to the spread and/or the severity of Covid19, through two possible mechanisms:

- Acute: microparticles derived from fossil fuels might act as airborne carriers of virus;
- Chronic: exposure to microparticles and chemical pollutants might cause chronic lung injury, exacerbating the consequences of viral infection.

The strategy of the project is to develop a ML-based study to evaluate a possible statistical association between air pollution and Covid19 including also a wide variety of components that are expected to influence rates of SARS-COV-2 diffusion and infection. The latter is one of the key features of PLANET, this is why quite some effort has been made in order to collect heterogeneous data such as: atmospheric data, population density, urban vs rural environment, mobility, socio-economic conditions. To this end, data from several sources have been collected, namely: data from the Regional Agency for Environment Protection (Agenzia Regionale per la Protezione dell'Ambiente, ARPA) of Umbria region [3] (both sensors and models); both data from Copernicus Atmosphere Monitoring Service (CAMS) and Climate data [4]; the Italian National Institute of Statistics (Istituto Nazionale di Statistica, ISTAT) [5]; Deprivation Indexes; Epidemiological data. The latter currently includes: Data from the Italian National Institute of Health (Istituto Superiore di Sanità, ISS); Data from the Local Sanitary Agency (Azienda Sanitaria Locale, ASL) of Viterbo (IT); Data from Liguria region. Personal data received by ISS and ASL have been managed in the context of the organizational framework of the EPIC cloud, which ensures that the right agreements are in place in order to comply with GDPR as both Data Controller (ISS data) and Data Processor (ASL Viterbo data). All this variety makes PLANET a perfect occasion for INFN to develop a technical solution which is possibly generic, extensible and reusable. To this end a key aspect consists in being based on cloud-native paradigm, following the main assets of INFN Cloud project which represents the driving force for the Cloud development of all INFN initiatives.

## 3. A Cloud native platform for heterogeneous data handling

A requirement analysis has been performed in order to identify the computing needs of the PLANET project. One of the first key items that emerged is that the heterogeneous data acquired should be kept and stored in its native format. Ideally data should remain in this condition until it is necessary to define a structure for the related analysis. Analysts would like any data scheme to be defined at analysis time rather than at archiving time due to the intrinsic heterogeneity of the treated dataset. This requirement is also imposed by the data driven nature of the analysis strategy where one needs to be able to come back and reprocess data at any time. Indeed, fixing any schema upfront might result in loosing the opportunity to later explore new ideas. In addition the possibility to retrieve data by type, find their location in the archive and possibly download it for a quick inspection, translates in the need for some kind of catalogue avoiding data swamps scenarios.

Interfaces for any aspect of the data manipulation should be friendly and easy to learn. Indeed the objective of the project should be focused on getting data insights and not to deal with technical

configuration which might become obstacles on the analysis path. Also, given the ecological nature of the project, new data sources can be identified along its lifetime, thus new source of information might be added at any given point in time. Being able to reprocess automatically new data, through the support of a workflow automation infrastructure, represents a great added value.

Eventually the request to be able to support both structured and unstructured data emerged in order to extend the study to new sources. In summary we identified few key items to be taken into account for the design of our solution:

- To use open source solutions that provides a sustainable longer term maintenance strategy
- To minimize in-house developments, limited to integration interfaces if needed
- To develop a clear and simple design in order to allow scalability and to ease operations
- To support automation and self-healing solutions. The system should react based on events and/or manual triggers
- To adopt storage based solution for data archival in view of unstructured data and to avoid multiple databases synchronization
- To enhance the use of metadata enabling FAIR [6] (Findable, Accessible, Interoperable, Reusable) data management.

### 3.1 A storage centric design

Object storage technology [7], nowadays, is the most popular option in a cloud environment, due to its versatility and its simplicity to integrate with other systems, along with a battle-tested scalability. MinIO [8] is an open source object storage service, providing an "AWS S3-compatible" storage. We decided to chose it as a core component to build our the cloud native solution. The S3 compliance of MinIO grants the possibility to support things like the use of [9], as well as to rely on AWS Security Token Service (STS) [10] protocol for authentication (*AuthN*) and authorization (*AuthZ*). The latter allowed us to combine native support for external OIDC [11] identity providers, an important feature since it represents a strong requirement in order to support sensitive data archival and processing. The powerful WebUI (provided out-of-the-box) is surely an added value both from operational perspectives as well as from the end user point of view. Among others, the MinIO's capabilities we identified to be particularly relevant to us are:

- Bucket notifications that allows to send events to supported external services on specific object or bucket events
- Support for customizable authorization policies webhook, like the one provided by OpenPolicyAgent service
- Metadata: it writes and operates on metadata and data together providing a granularity at the level of individual objects [12]

Indeed the metadata handling represents one of the most important aspects in the design we propose. Although the default set of metadata is managed only by the MinIO service, thus not

very useful for the end-users needs, the possibility to enrich the file uploaded with additional user defined metadata and tags are also supported. The enrichment process of course is completely up to the users and, as a consequence, it allows to support any custom use case as the user can define metadata for any purpose. We firstly focused on a simple example to validate the enrichment process enforcing the enrichment with a minimal set of information in order to know which data are in the archive at any point in time, thus to be able to retrieve files and/or to filter them given a specific data type. In such a way we effectively uses MinIO not only as a data storage but also as a metadata manager, avoiding the need to use any external service, and to rely on S3 APIs to search and inspect the buckets before a real file download. We adopted this approach in order to enable the possibility for the user to specify extra information as metadata and to upload them during the data injection as shown in Fig. 1 To this end we developed a Go [13] component with a very basic GUI that ease

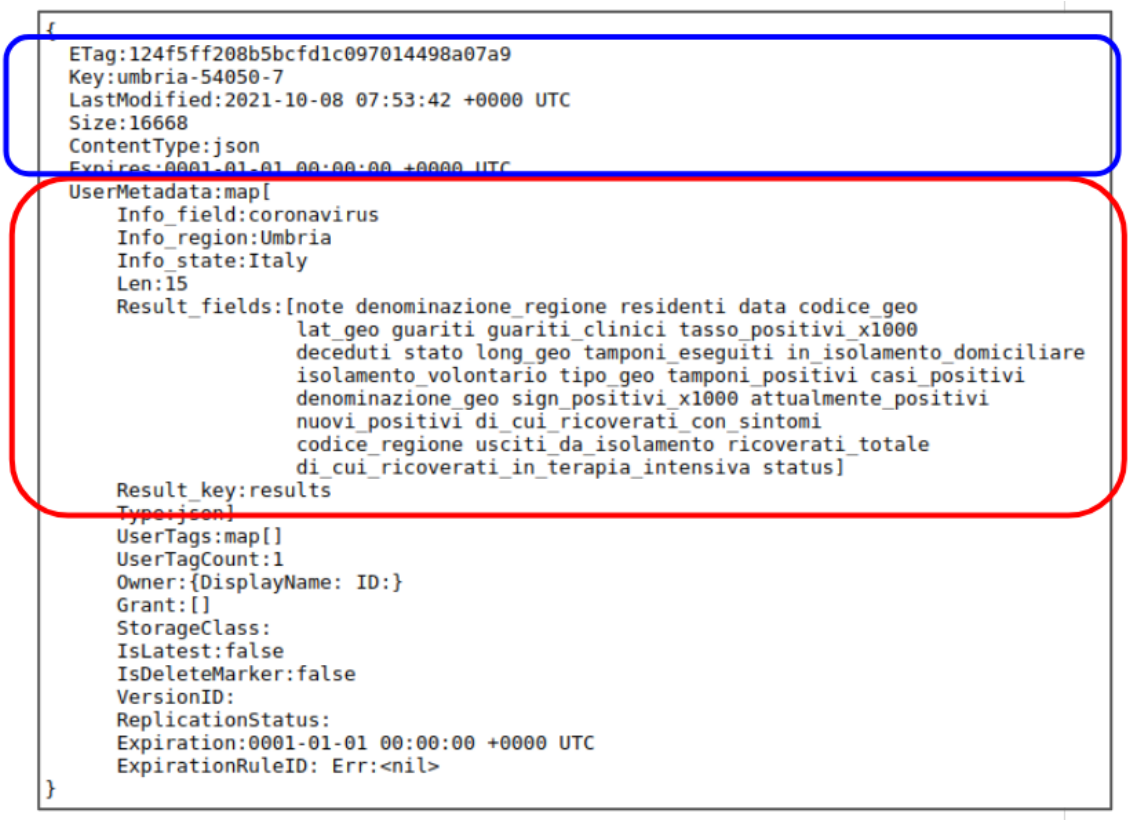


**Figure 1:** Simple example of metadata enrichment.

the process of extending metadata while uploading file. This was a pre-requisite to actually enable the proposed injection flow. In this scenario the user will upload any file to a specific bucket (i.e. RAW data Bucket), then, data will be processed and validated (meaning that for each uploaded file the related metadata is verified). Upon success the file is moved to the Validated repository and made available for the users for further processing. In case of the metadata check failure, the file is moved into a Triage bucket, a sort of limbo where the files are not completely available because they lack associated metadata. It is important to note as validation process is based on checks that the users can define and modify at any point in time. The solution is completely agnostic of the

specific type of file, and the files in the limbo can be re-checked and the related metadata can be fixed. As anticipated in Sec 3 the goal is to fully automate the validation processes, meaning that we do not expect a per-file check to be performed by humans. Humans need just to declare which kind of check is needed for a given use case and/or category of data, and, obviously, to ensure the proper metadata enrichment. In this respect the bucket notifications support of MinIO become the technology enabler.

### 3.2 Enabling PLANET pipelines

Stored data can be friendly enriched with user level metadata. The MinIO services support bucket notification which might be used as a trigger for further actions. What we needed then is an event-driven automation and, at least, a simple web service to make the metadata usable by the end-users. To cope with the automation requirement we decided to use Argo Workflows [14], an open-

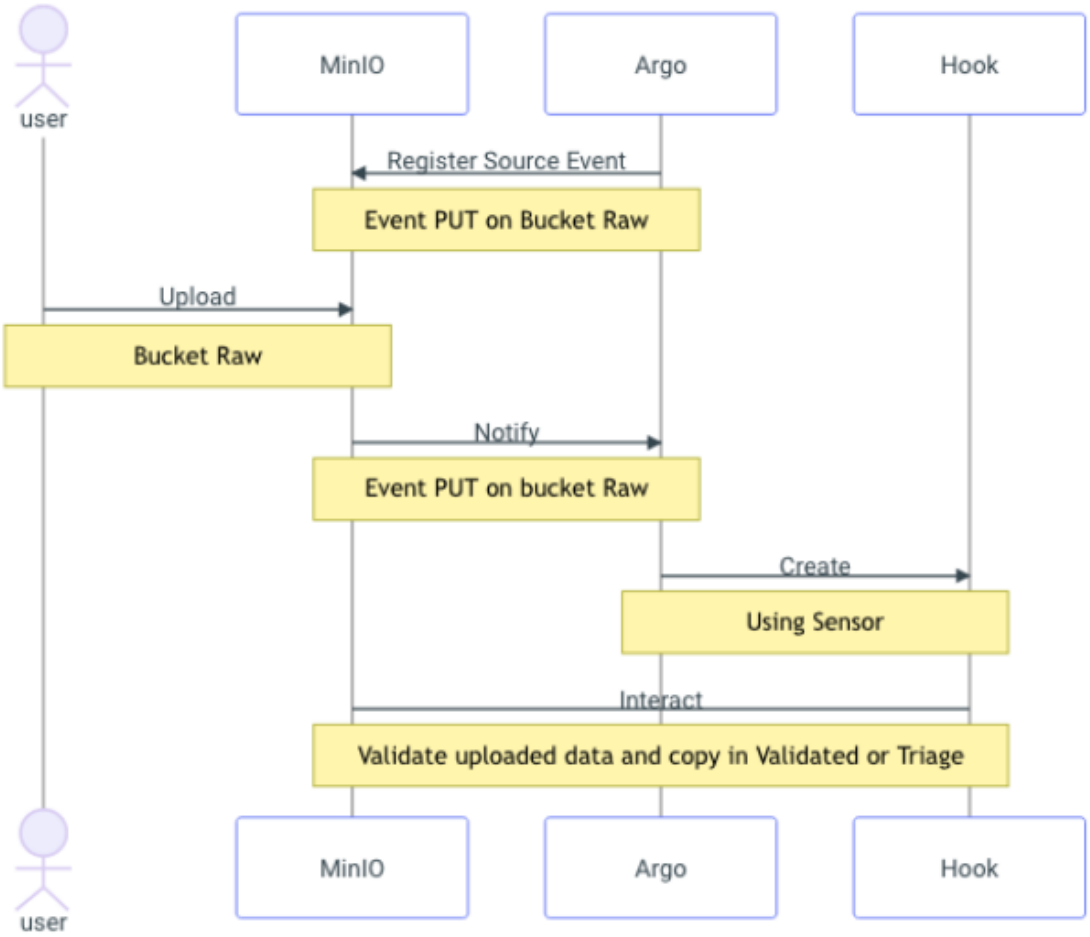


**Figure 2:** Flow of the interactions between MinIO and Argo implemented in PLANET to support the automated metadata validation.

source container-native workflow engine for orchestrating complex workflows on Kubernetes [15] with a declarative approach. Also, it is capable of binding the execution of a configured workflow

to the reception of different kind of events. The framework providing this functionality is named Argo Events. In other words it is possible to register a reaction to a specific event coming, for instance, from the MinIO bucket notification system. To summarize, it is possible to create a trigger for a particular event that happened on MinIO such as the arrival of new data in a bucket, thus we decided to implement the workflow as reported in figure 2. Argo listens to a specific event on the MinIO storage (*eventName: 'ObjectCreated:Put'*) that will trigger the workflow execution through an Argo hook called "sensor". Such a hook is the heart of the flow, it is where the real interaction with the object storage happens. The hook is de-facto a containerized application which can be any user defined action coded in any preferred language. In our case we developed a simple Go program in charge of validating the data inserted by the user in the bucket Raw and contained in the message sent by the MinIO notification. The check is performed against few key parameters that we configured all the possible metadata as compulsory. Also, the hook is responsible for moving data either into the Validated or the Triage bucket, depending on the result of the validation process. All of these actions do not require any human action other then the metadata enriched data injection following the rules that should match the validation check. Users will be notified when data goes into the triage since at that point there are actions required to fix the inconsistency. One of the main reason in order to implement this metadata enrichment, and the related validation, has been to allow data findability. As discussed in section 3, indeed the user should be capable to explore the content of the archive and possibly find the data to retrieve it or just to access and inspect it. To achieve this objective we developed a lightweight web service which implements the flow described in the schema as reported in fig. 3.


**Figure 3:** Workflow of the metadata interaction implemented by the custom web-app

At first, after the user login, there is an automatic metadata indexing occurring at the service level, than it will enable an efficient search procedure. Also, this can be re-triggered by the user

itself or another event managed by Argo, e.g. when a new file is uploaded or at the end of the day. The collector is the entity responsible of the storage interaction, for retrieving user capability information, and the actual metadata collections, no other repositories or databases are involved. When the user will select specific metadata, the collector will then respond with the file list that satisfies the metadata filter.

### 3.3 A multi user analysis environment for data analysis

Once data are injected, the last requirement is to allow an easy-to-use interface to inspect and to analyse the datasets. A rather obvious decision made in this respect has been to provide a JupyterHUB [16] instance properly configured to manage user authentication and authorization, based on external OIDC IDP such as INDIGO-IAM [17]. The PLANET project deployed a dedicated instance of such an identity provider (i.e., https://planet-iam.cloud.cnaf.infn.it/) which is used by all the deployed services in the stack as shown in figure 5. In this way we provided a multi-user JupyterHub instance which is able to support the on-demand spawning of customizable UIs, and which allow a user level definition of the runtime environment requirements. In other words, every user can customize the UI based on the analysis specific needs. A utility named *sts-wire* [18] has been developed in order to provide the user with a POSIX-like [19] experience while accessing data stored on the object storage. It is meant to allow an easy and authenticated (and authorized) to explore bucket contents, not to support any highly I/O data processing, but rather to facilitate a first content insight. From the technical perspectives it is based on the following key elements:

- oidc-agent [20]: a software allowing to manage the lifetime of OIDC tokens ( JSON Web Token [21]. ) that is required for a secure authentication with the remote services.
- AWS STS *AssumeRoleWithWebIdentity*: a protocol APIs to allow the creation of temporary credentials exchanging JWT returned from a configured OpenID Identity Provider
- rclone: a command-line program to manage files on many cloud storage, including S3, and with the ability to mount the contents as a virtual filesystem

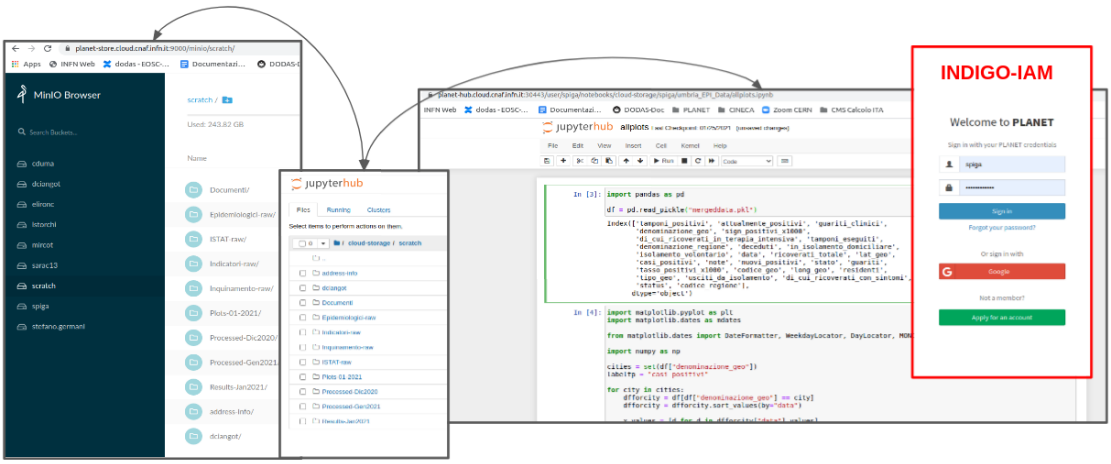


**Figure 4:** Integrated jupyter hub system for data access and data analysis

*sts-wire* has been fully embedded in the Jupyter environment by meaning that any JupyterLab instance, that is automatically spawned, handles the installation and the configuration of the *sts-wire* against the planet dedicated services. Also it automatically setup oidc-agent accounts and keep refreshing token. All this process does not require any human action, beside the request to spawn the server. Finally *sts-wire* provides the support for accessing data from any user remote work station.

## 4. Status of the architecture

The system described in the section 3 has been fully integrated, deployed, and made available to the PLANET project. It is what the project named PLANET DataLake. Technically wise, all the services are deployed over a Kubernetes dedicated instance as shown in figure 5. Helm charts [22] have been developed so that the complete platform deployment is completely automated and easy to replicate anywhere via a declarative configuration, enabling an easy adoption also for other projects, one of our initial aim. All the stack has been integrated with the INDIGO-IAM and, as a matter of fact, it represents a SSO experience. Any exposed front-end, being it for data injection, data analysis or to simply find the data, follows the very same pattern for what concern authentication and authorization.

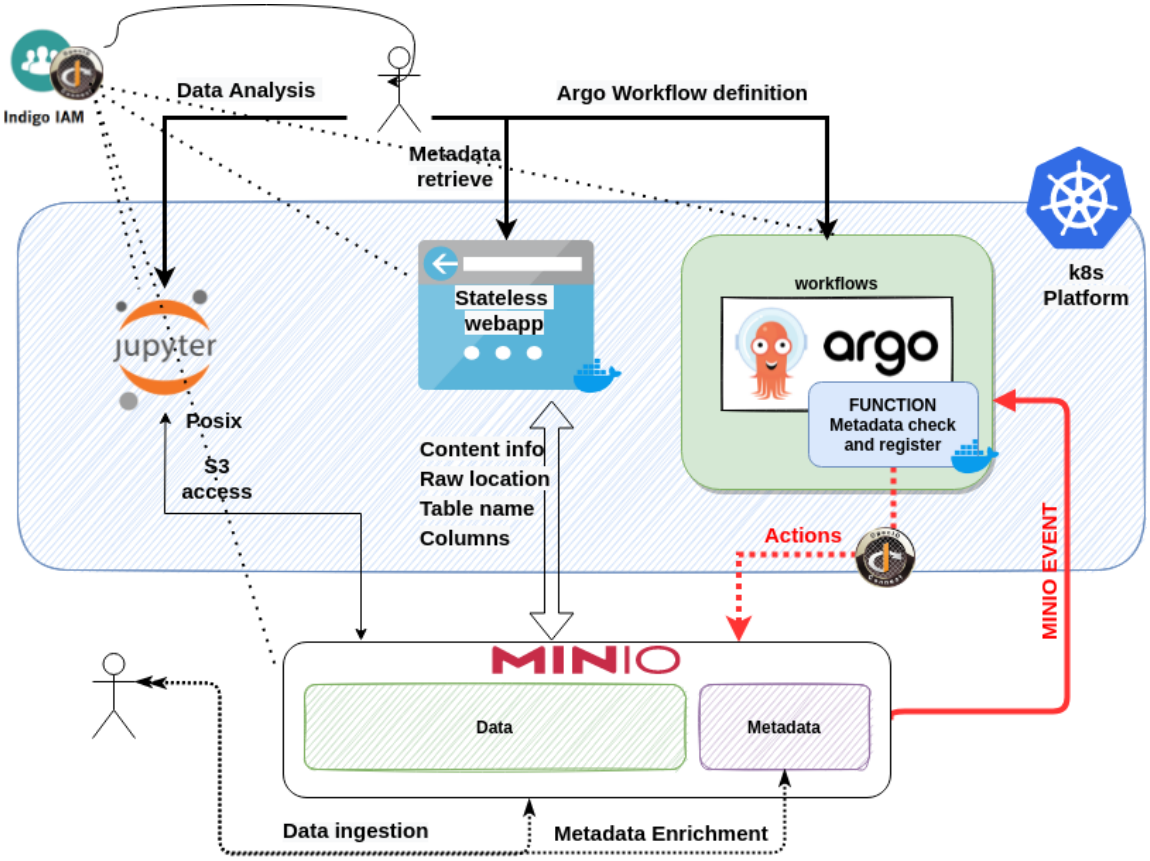


**Figure 5:** A complete high level schema of the PLANET Architecture

## 5. Support to sensitive data through the INFN EPIC-Cloud

As anticipated one of the challenge we wanted to address is the sensitive data handling, being PLANET in the conditions of treating medical data. The General Data Protection Regulation (GDPR) is in force since 2018 and mandates to apply the principles of lawfulness; fairness and transparency; purpose limitation; data minimisation; accuracy; storage limitation; integrity and confidentiality; and accountability. A strong governance structure is essential to standardise privacy and to develop privacy by design and default as required by the Regulation. For this reason, we decide to adopt the EPIC Cloud (Enhanced PrIvacy and Compliance): a partition of INFN Cloud certified ISO/IEC 27001. EPIC is not only a provider of resource, but mainly a set of well defined technical and organizational security measures, with well defined structures and responsibilities. More in detail, EPIC consists of an OpenStack instance and of a set of ancillary services managed in the context of an Information Security Management System (ISMS), which enables us to guarantee the adoption of high security standards and to guarantee the compliance to GDPR and other privacy laws and regulations. In EPIC the security posture is stronger than the one normally provided to HEP researcher. The EPIC ISMS is implemented in accordance with the ISO/IEC 27001 guidelines. Moreover, it complies with the ISO/IEC 27017 ( i.e., Code of practice for information security controls based on ISO/IEC 27002 for cloud services), and with the ISO/IEC 27018 (i.e., Code of practice for protection of personally identifiable information (PII) in public clouds acting as PII processors).

Among the organizational measured adopted in EPIC it worth mentioning:

- the definition of a clear organizational structure, with specification of accountability and responsibility (RACI matrices).
- the definition and documentation of a clear and measured organizational processes covering information security incident management; risk management; compliance management; personnel competence, skills and awareness management.
- the adoption of a robust authentication/authorization system providing 2-factor authentication, segregation of duties and account lifecycle management.
- the adoption and enforcement of the *Least Privilege* policy.
- the adoption of the *Privacy by design* methodology, which can be summarized by the following principles: Proactive not reactive; preventive not remedial; Privacy as the default setting; Privacy embedded into design; Full functionality positive-sum, not zero-sum; End-to-end security-full lifecycle protection; Visibility and transparency -keep it open; Respect for user privacy -keep it user-centric [23].
- Encryption at-rest (backup) and in-transit.

At the time of writing we actually started the migration process, that is, the platform described in section 4 is moving in to the EPIC system. The compute part has been fully implemented as show in the figure 6. This is the first step toward the full migration, where all the stack will be hardened applying the technical measures required to improve the security of the whole system.

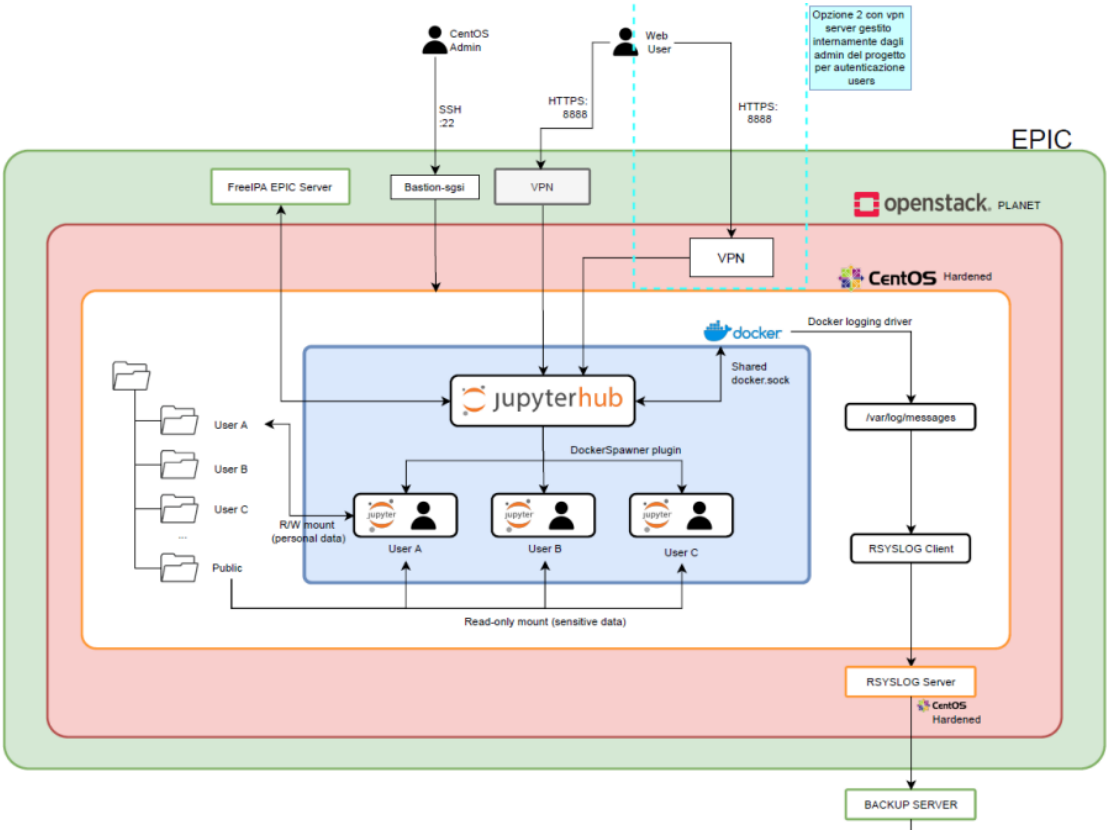


**Figure 6:** Schema of the current EPIC based platform for PLANET project

## 6. Summary and future

PLANET is a multidisciplinary project aiming at analyzing heterogeneous and sensible data. In order to enable effective data processing, a generic cloud-native platform has been successful prototyped and made available at INFN Cloud. The developed system is highly centred on metadata management and enrichment, this guided several technical choices during the design phase. The event based system for the workflow automation represents a key element, and the early adoption resulted very promising with no show stoppers. Further enhancements are expected and a concrete example is the automation of data reprocessing. There is an ongoing activity to move the PLANET platform inside the EPIC-Cloud environment, with the main objective to harden the current platform applying the technical measures required to improve the security of the whole system. We envision this as a very generic solution, and we foreseen that this will be integrated and made available as an "open" and generic and reusable platform EPIC compliant within the INFN Cloud ecosystem. This would be the natural evolution toward an integration of data from multiple information sources, as well as to support descriptive, predictive and real time analysis.

## References

[1] INFN Cloud - Cloud resources for research. URL https://www.cloud.infn.it/

[2] A. Chierici, B. Martelli et al; SGSI project at CNAF. EPJ Web of Conferences 214, 08017 (2019), DOI 10.1051/epjconf/201921408017

[3] https://www.arpa.umbria.it/

[4] https://www.copernicus.eu/en/accessing-data-where-and-how/conventional-data-access-hubs

[5] https://www.istat.it/en/

[6] Wilkinson M.D. Dumontier M. Aalbersberg I.J. Appleton G. Axton M. Baak A. et al. The FAIR Guiding Principles for scientific data management and stewardship. Sci Data. 2016; 3https://doi.org/10.1038/sdata.2016.18

[7] M. Factor, K. Meth, D. Naor, O. Rodeh and J. Satran, "Object storage: the future building block for storage systems," 2005 IEEE International Symposium on Mass Storage Systems and Technology, 2005, pp. 119-123, doi: 10.1109/LGDI.2005.1612479.

[8] Multi-Cloud Object Storage https://min.io/

[9] Boto3 - The AWS SDK for Python https://github.com/boto/boto3

[10] Security Token Service https://docs.aws.amazon.com/cli/latest/reference/sts/

[11] OpenID Connect, https://openid.net/specs/openid-connect-core-1_0.html

[12] https://blog.min.io/minio-versioning-metadata-deep-dive/

[13] The Go Programming Language. URL https://go.dev/

[14] https://argoproj.github.io/

[15] https://kubernetes.io/

[16] A multi user version of the notebook designed for companies, classrooms and research labs; https://jupyter.org/hub

[17] Andrea Ceccanti, Enrico Vianello, & Marco Caberletti. (2018, May 18). INDIGO, Identity and Access Management (IAM) (Version v1.4.0). Zenodo. DOI: 10.5281/zenodo.1874791

[18] sts wire package: https://github.com/DODAS-TS/sts-wire

[19] POSIX.1 2008 (IEEE Std 1003.1 2008), http://pubs.opengroup.org/onlinepubs/9699919799/

[20] Zachmann, G., (2018). OIDC-Agent: Managing OpenID Connect Tokens on the Command Line. In: Becker, M. (Hrsg.), SKILL 2018 - Studierendenkonferenz Informatik. Bonn: Gesellschaft für Informatik e.V.. (S. 11-21).

[21] JSON Web Tokens, RFC 7519, https://tools.ietf.org/html/rfc7519

[22] Helm - The package manager for Kubernetes. URL https://helm.sh/

[23] Cavoukian, A. Privacy by design: the definitive workshop. A foreword by Ann Cavoukian, Ph.D. IDIS 3, 247–251 (2010). https://doi.org/10.1007/s12394-010-0062-y