Ref. Ares(2026)4371487 - 28/04/2026

NOSTRADAMUS

Deliverable D2.3

D2.2 - Software Architecture for Cloud, Edge and Mixed Solutions

Funded by the European Union

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Research Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

NOSTRADAMUS

PARTNERS

ERATOSTHENES CENTRE OF EXCELLENCE

DLR Deutsches Zentrum für Luft- und Raumfahrt German Aerospace Center

AgFutura TECHNOLOGIES

CROPT

ITC INOVACIJSKO TEHNOLOŠKI GROZD INNOVATION TECHNOLOGY CLUSTER

AGRIFOOD DIGITAL INNOVATION HUB

DLG

CZU Czech University of Life Sciences Prague

BioSense INSTITUTE

TERRAQUE Advancing Earth Science

Kmetijsko gozdarska zbornica Slovenije
KMETIJSKO GOZDARSKI ZAVOD MURSKA SOBOTA

FGS

AGRICULTURAL RESEARCH INSTITUTE

ARISTOTLE UNIVERSITY OF THESSALONIKI

UNIVERSITÉ DE GENÈVE
INSTITUTE FOR ENVIRONMENTAL SCIENCES

University of St.Gallen

Agroscope

NOSTRADAMUS

Grant Agreement No.: 101134888
Call: HORIZON-CL6-2023-GOVERNANCE-01
Topic: HORIZON-CL6-2023-GOVERNANCE-01-13
Type of action: HORIZON Research and Innovation Actions

D2.3
D2.2 - SOFTWARE ARCHITECTURE FOR CLOUD, EDGE AND MIXED SOLUTIONS

Work package	2
Task	2.3
Due date	30/04/2026
Submission date	28/04/2026
Deliverable lead	Aristoteleio Panepistimio Thessalonikis (AUTh)
Version	1.0.1
Authors	Konstantinos Panayiotou (AUTH), Gregory Giuliani (UNIGE), Pedro Pereira Gonçalves (TERRADUE)
Reviewers	Themistoklis Diamantopoulos (AUTH), Daria Loginova (HSG), Stelios Neophytides (ECoE), Ioannis Varvaris (ECoE), Thomaida Polydorou (ECoE)
Abstract	This Deliverable (D2.3, early version of D2.2 - Software Architecture for Cloud, Edge and Mixed Solutions) provides the comprehensive technical blueprint for the Nostradamus digital ecosystem, establishing a modular and scalable hybrid Edge-to-Cloud architecture specifically engineered to manage the "four V's" of agricultural Big Data: Volume, Variety, Velocity, and Veracity. The document details the integration of the platform's four architectural pillars—the Data Hub (I), the IoT Observatory (II), the Low-Code Platform (III), and the Data Cubes (IV), fusing Lambda and Kappa patterns to reconcile high-velocity real-time event streaming with high-precision historical analysis. A core innovation presented is the decomposition of cyber-physical applications into three interoperable functional tiers (App-Edge, App-Logic, and App-GUI) orchestrated through a Low-Code Platform that utilizes Model-Driven Engineering and specialized Domain-Specific Languages. By incorporating an AI Assistant for describing applications via natural language, the architecture democratizes

	development for experts while providing a robust foundation for the implementation of the project's four core agricultural modules and the subsequent WP6 Open Calls.
Keywords	Architecture, Edge, Cloud, Big Data Management, IoT, Applications

Document Revision History

Version	Date	Description of change	List of contributors
0.1.0	01/04/26	First draft	Konstantinos Panayiotou
0.2.0	13/04/26	General contribution Adds content in sections 4 & 5	Gregory Giuliani
0.3.0	17/04/26	General contribution. Adds content in sections 4 & 5	Pedro Pereira Gonçalves
0.4.0	20/04/26	Revised document	Konstantinos Panayiotou
	21/04/26	Document Review	Daria Loginova
0.5.0	22/04/26	Revised document	Konstantinos Panayiotou
0.6.0	25/04/26	Document Review	Stelios Neophytides
1.0.0	25/04/26	Final Version	Konstantinos Panayiotou
1.0.1	26/04/26	Final Review, Formatting Revisions	Thomaida Polydorou, Ioannis Varvaris

DISCLAIMER

Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Health and Digital Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

COPYRIGHT NOTICE

© NOSTRADAMUS Consortium, 2026

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

appropriate citation, quotation, or both. Reproduction is authorized provided the source is acknowledged.

The (insert) Consortium is the following:

Participant number	Participant organisation name	Short name	Country
1	ERATOSTHENES CENTRE OF EXCELLENCE	ECoE	CY
2	DEUTSCHES ZENTRUM FUR LUFT – UND RAUMFAHRT EV	DLR	DE
3	AG FUTURA TECHNOLOGII DOOEL SKOPJE	AGFT	MK
4	CROPT DOO NOVI SAD	CROPT	RS
5	ITC - INOVACIJSKO TEHNOLOSKI GROZD MURSKA SOBOTA	ITC	SI
6	DLG EV	DLG e.V.	DE
7	CESKA ZEMEDELSKA UNIVERZITA V PRAZE	CZU	CZ
8	BIOSENSE INSTITUTE – RESEARCH AND DEVELOPMENT INSTITUTE FOR INFORMATION TECHNOLOGIES IN BIOSYSTEMS	BIOS	RS
9	TERRADUE SRL	TERRADUE	IT
10	KMETIJSKO GOZDARSKA ZBORNICA SLOVENIJE KMETIJSKO GOZDARSKI ZAVOD MURSKA SOBOTA	KGZS – ZAVOD MS	SI
11	F6S NETWORK LIMITED	F6S	IE
12	MINISTRY OF AGRICULTURE, RURAL DEVELOPMENT AND ENVIRONMENT OF CYPRUS	MARDE	CY
13	ARISTOTELIO PANEPISTIMIO THESSALONIKIS	AUTH	EL
14	UNIVERSITE DE GENEVE	UNIGE	CH
15	UNIVERSITAET ST. GALLEN	HSG	CH
16	AGROSCOPE	AGS	CH

EXECUTIVE SUMMARY

The NOSTRADAMUS initiative, funded by Horizon Europe, addresses critical challenges in food security, sustainability, and agricultural resilience within the European Union by establishing a robust, data-centric foundation for agricultural independence. The project's overarching goal is to design, develop, and evaluate a consolidated, scalable, and user-friendly data silo capable of online and real-time harvesting of valuable information from heterogeneous sources to transform the EU agricultural sector.

This deliverable, numbered D2.3 (early version of Deliverable "D2.2 - Software Architecture for Cloud, Edge and Mixed Solutions"), presents the comprehensive technical blueprint for the NOSTRADAMUS digital ecosystem, specifically detailing the software architecture for cloud, edge, and mixed solutions. It is strategically positioned to bridge the gap between initial requirement analysis, established in D2.1, and the subsequent development of prototypes, data pipelines, and end-user applications. The architecture defines the structural rules, component interactions, and the "Edge-to-Cloud" logic that will underpin the entire system, transforming user-centric data needs into a robust "System of Systems" architectural design.

The proposed architecture is built upon a hybrid Edge-to-Cloud model and guided by principles of modularity, microservices, open standards, and interoperability to prevent vendor lock-in and ensure wide adoption. It comprises three distinct layers:

1. Data Infrastructure Layer: This layer serves as the "Batch Layer" and long-term storage engine, hosting the Data Hub and Data Cubes. It is responsible for managing massive static datasets, Earth Observation (EO) archives, and socio-economic indicators, providing multi-dimensional arrays for harmonizing geospatial data.

2. IoT Observatory Layer: Functioning as the "Speed Layer," this specialized Big Data Management System (BDMS) is engineered for real-time event handling. It ingests and processes high-velocity sensor data streams with sub-second latency, utilizing distributed technologies like Apache Kafka and Apache Flink.

3. Low-Code Application Layer: This layer acts as the democratization interface, providing tools for experts, such as SMEs, Agri-industry, Cooperatives, Agri-food companies, or even scientists to build custom digital solutions, as third parties successfully responding to the NOSTRADAMUS Open Calls. Leveraging Model-Driven Engineering and Domain-Specific Languages (DSLs), an Application Generation Engine automatically transforms description models into executable software artifacts in the form of applications, alleviating the technical burden of direct programming. On the other hand, farmers, agronomists, and policymakers will make use of these NOSTRADAMUS-powered Applications.

The architecture addresses functional requirements across several domains, including data ingestion from heterogeneous sources (e.g., high-velocity IoT streams, massive Earth Observation files), data processing (real-time and batch), data management & storage (e.g., Cassandra for IoT,

Object Storage or geospatial), application generation (for edge components, application logic, and graphical user interfaces), and data access & visualization. Non-functional requirements emphasize performance, scalability (e.g., Kubernetes for orchestration), reliability, security (e.g., Keycloak for authentication/authorization), and interoperability (e.g., OGC/ISO standards, CWL pipelines for reproducibility).

This architectural design, spearheaded by AUTH, facilitates the rapid deployment of data-driven digital solutions across five demonstration sites located in Germany, Serbia, Switzerland, Slovenia, and Cyprus. It supports core modules such as drought monitoring, crop suitability, soil fertility, and sustainable pest management, to store and retrieve information from heterogeneous data sources, such as EO data, sensory IoT data from the edge and socioeconomic datasets. Moreover, it serves as the foundation for the Application Generation Engine (D2.3 in its earlier iteration of the project, which is now D2.5/D2.6 deliverables as further iterations of the project), the Data Hub operation platform (D3.2), Geodatabase and Data Cube Infrastructure (D5.1), and enables third parties to develop interoperable digital applications through Open Calls (WP6).

By providing this robust, scalable, and user-friendly framework, NOSTRADAMUS aims to equip stakeholders with actionable insights to optimize operations, improve yields, and reduce agricultural inputs, thereby boosting EU resilience, independence, and ecological ambitions for agriculture, including resource-constraint environments.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

API	Application Interface
CPS	Cyber-Physical System
DC	Data Cube
DSL	Domain-specific Language
EO	Earth Observation
IoT	Internet of Things
IP	Internet Protocol
IT	Information Technology
LCP	Low-code Platform
MDE	Model-driven Engineering
TCP	Transmission Control Protocol
UI	User Interface

1 INTRODUCTION

The NOSTRADAMUS initiative is designed to address the critical challenges of food security, sustainability, and agricultural resilience within the European Union. Funded by Horizon Europe, the project aligns with key EU frameworks such as the European Green Deal and the Common Agricultural Policy to establish a robust, data-centric foundation for EU agricultural independence. The primary objective is to design, develop, and evaluate consolidated, scalable, and user-friendly data silo, the Data Cube Ecosystem, capable of online and real-time harvesting of valuable information from heterogeneous sources.

To achieve these objectives, NOSTRADAMUS introduces a **hybrid Edge-to-Cloud architecture** divided into three distinct layers:

1. **Data Infrastructure:** Provides access to various data sources, such as EO, socio-economic, and Data Cubes.
2. **IoT Observatory (IoTO):** Built on top of state-of-the-art tools (e.g., Kafka, Flink) for near real-time IoT data management, IoTO acts as the gateway for edge and IoT devices to connect, send and aggregate data via a unified API.
3. **Low-Code Platform (LCP):** Facilitating the development of open-source digital applications by domain experts without technical and technical backgrounds.

This technical framework is validated through a practical implementation strategy. The project encompasses **five demonstration sites** (located in Germany, Serbia, Switzerland, Slovenia, and Cyprus), situated across a range of biogeographical and socio-economic regions. These sites will serve not only as pilot locations for the refinement and testing of the project's **Core Modules**, specifically drought monitoring, crop suitability, soil fertility, and sustainable pest management, but also as multifaceted assets. They will be used to demonstrate digital applications, facilitate training activities, and host visits with young scientists, IT experts, and farmers' associations. Furthermore, **third parties** funded through Open Calls may potentially utilize these pilot sites to develop and exhibit their own digital applications. These agricultural fields will facilitate collaborations in the context of the NOSTRADAMUS project, fostering a **Multi-Actor Approach (MAA)** that connects IT providers, policymakers, and land managers in a continuous co-creation process.

1.1 RELATION TO OTHER DELIVERABLES

This deliverable D2.3, the early version of **D2.2 - Software Architecture for Cloud, Edge and Mixed Solutions**, serves as the comprehensive technical blueprint for the NOSTRADAMUS digital ecosystem. It is strategically positioned to bridge the gap between the initial requirement analysis and the subsequent development of prototypes, data pipelines, and end-user applications

Specifically, the relationship between D2.3 and other project deliverables is defined as follows:

- **Inputs from D2.1 Needs-based data requirements**: The architectural design presented in this document is directly derived from the functional and non-functional requirements established in D2.1. It addresses the specific data collection, processing, and storage needs identified for the five demonstration sites and the project's core modules.
- **Foundation for D2.3 Application Generation Engine:** This architecture defines the structural rules, component interactions, and Edge-to-Cloud logic that will be automated by the Application Generation Engine. While this deliverable specifies how the components (Edge, App Logic, and GUI) interact, D2.3 will provide the bootstrapping software and Domain Specific Languages (DSLs) to allow third parties to generate these components automatically.
- **Framework for D3.2 Data Hub operation platform (Algorithm Integration & Operations):** D2.2 establishes the cloud-native standards required for Task 3.5 (Algorithm integration and packaging) and Task 3.6 (Platform operations). Specifically**:**
    - Algorithm Packaging: Defines the containerization standards (Docker) and workflow specifications (Common Workflow Language - CWL) that D3.2 must implement to allow researchers to package their algorithms for the platform.
    - Platform Operations: The hybrid infrastructure and orchestration layers (Kubernetes) defined in this architecture provide the technical foundation for the operational environments managed in D3.2, ensuring the system can handle data ingestion and processing pipelines at scale.
- **Infrastructure for D5.1 Geodatabase and Data Cube Infrastructure:** While the current document (D2.2) establishes the high-level interoperability standards and the position of the Data Cubes within the "Data Infrastructure" layer of the hybrid architecture, D5.1 provides the concrete technical environment and infrastructure selection required to host them. Specifically, D5.1 details the output of Task 5.1 and Task 5.3, defining the selection of the cloud infrastructure provider and the specific software stack for the "***D****ata Cube on Demand*" (DCoD) implementation**.**
- **Enabler for D6.2 Digital applications developed by Third Parties:** By establishing a standardized architecture for cloud, edge, and mixed solutions, this deliverable creates the technical "playground" for the Open Calls in WP6. It ensures that applications developed by funded third parties (D6.2) are interoperable with the NOSTRADAMUS ecosystem and can effectively utilize the core modules and data silos.

In summary, D2.3 (and its update D2.4 to be submitted in M) acts as the central reference point that harmonizes the data requirements (D2.1 & D2.2) with the operational platforms (D3.2, D5.1) and the application development tools (D2.5 & D2.6), ensuring a cohesive "*System of Systems*" approach.

1.2 DOCUMENT STRUCTURE

The present document is structured to provide a logical progression from the high-level project vision and stakeholder requirements to the detailed technical specifications of the Nostradamus "System of Systems." The structure is organized as follows:

- **Section 1 – Introduction:** Defines the scope of the deliverable, its relation to other project tasks, and the primary objectives of the software architecture.

- **Section 2 — Software Requirements:** Outlines the functional and non-functional requirements categorized by Volume, Variety, Velocity, and Veracity, establishing the technical constraints for the architectural design.
- **Section 3 — Overall System Architecture:** Establishes the project's Edge-to-Cloud vision, defines the Stakeholder Archetypes (Technical Intermediaries vs. Domain Experts), and introduces the conceptual model for Hybrid Applications.
- **Section 4 — Architectural Pillars:** Provides a comprehensive deep dive into the four core pillars of the platform:
    - 4.1 Data Hub (I): Architecture for EO data lifecycle management.
    - 4.2 IoT Observatory (II): Specifications for the speed layer and event-streaming (Kafka/Cassandra).
    - 4.3 Data Cubes (IV): Implementation of multi-dimensional arrays for spatiotemporal analysis.
    - 4.4 Low-Code Application Platform (III): Details the Model-Driven Engineering (MDE) framework and the AI-assisted development workflow.
- **Section 5 — Data Pipelines:** Details the operational workflows for EO Data Pipelines and IoT Data Pipelines, specifying ingestion, transformation, and metadata registration (STAC) procedures.
- **Section 6 — Cloud, Edge, and Mixed Applications:** Presents the technical specification of the three-tier cyber-physical application model (App-Edge, App-Logic, and App-GUI) and the automation engine used for their rapid synthesis.
- **Section 7 — Infrastructure and Operations:** Details the robust, hybrid hardware and software foundation supporting the platform. This includes the specifications for the High-Performance Computing (HPC) environment at the Cyprus ECoE, and the implementation of cloud-native orchestration via Kubernetes. It also specifies the petabyte-scale storage strategies using S3.
- **Section 8 — Conclusions:** Summarizes the architectural achievements and outlines the roadmap for the subsequent development phases of the project.

This structure ensures that the deliverable serves as both a strategic blueprint for the consortium and a technical manual for third-party developers participating in the Nostradamus ecosystem.

2 REQUIREMENTS AND CONSTRAINTS

The system requirements defined in this section represent the operationalization of the user-centric needs previously identified in Deliverable D2.1. While D2.1 established the foundational data dependencies for the project's core modules, this section defines the functional behaviors and quality attributes the software architecture must support to manage those datasets at scale. To ensure high veracity and strategic alignment with EU data standards, every requirement is passed through a Validation and Prioritization Gating framework, ensuring that all technical specifications are measurable, actionable, and strictly adherent to the FAIR Principles.

2.1 SCOPE AND RELATION TO D2.1

While this deliverable (D2.3) focuses on defining the system-level technical requirements, encompassing the functional behaviors and non-functional quality attributes of the overall software architecture, it is essential to distinguish these from the data-specific requirements engineered earlier in the project. The needs-based data requirements were systematically gathered, analyzed, and documented in the previously submitted Deliverable D2.1. That foundational work identified the specific data types (EO data, IoT streams, socio-economic indicators), sources, and formats necessary to support the four core agricultural modules and the five European pilot sites.

Consequently, the requirements presented in this section of D2.3 represent the operationalization of those data needs into technical specifications. They define how the NOSTRADAMUS architecture must perform and interact to handle the high volume and high velocity data environments established in D2.1. Thus, D2.3 serves as the technical bridge, transforming the user-centric data needs of D2.1 into a robust, "System of Systems" architectural blueprint.

2.2 FUNCTIONAL REQUIREMENTS

The functional requirements for the NOSTRADAMUS software architecture are derived from the needs-based analysis of the pilot sites and the technical specifications of the three core architectural layers: the Data Infrastructure (Data Hub/Cubes), the IoT Observatory, and the Low-Code Platform. These requirements define the specific behaviors, functions, and interactions the system must support to enable the Hybrid Edge-to-Cloud model. They are categorized into five functional domains: Data Ingestion, Data Processing, Data Management & Storage, Application Generation, and Data Access & Visualization.

2.2.1 DATA INGESTION REQUIREMENTS

This category covers the system's ability to acquire data from heterogeneous sources, ranging from high-velocity IoT streams to massive Earth Observation (EO) static files. Data Ingestion requirements are summarized in Table 1.

TABLE 1: DATA INGESTION REQUIREMENTS

ID	Function Name	Subsystem	Description
FR1	**Concurrent Ingestion**	IoT Observatory	The infrastructure must support concurrent data ingestion from multiple heterogeneous sources (IoT devices, sensors) without performance degradation.
FR2	**Multi-Protocol Support**	IoT Observatory	The system must provide an Edge Gateway abstraction layer capable of ingesting data via standard IoT protocols, specifically MQTT and CoAP.

FR3	**Automated Schema Detection**	IoT Observatory	The system must support semi-automated schema identification to streamline data ingestion and ensure incoming events conform to expected formats.
FR4	**EO Data Acquisition**	Data Hub	The system must support the automated indexing of Earth Observation (EO) data products (e.g., Sentinel-1/2, Landsat) and their conversion into Cloud Optimized GeoTIFFs (COGs).
FR5	**CWL Ingestion Workflows**	Data Hub	All data ingestion processes must be defined as Common Workflow Language (CWL) pipelines to ensure the acquisition, conversion, and metadata extraction steps are reproducible and automated.

2.2.2 DATA PROCESSING REQUIREMENTS

These requirements define how the system transforms raw data into actionable intelligence using the Lambda/Kappa architecture principles defined in the architecture, and the functionality of the Data Hub. Data processing requirements are presented in Table 2.

TABLE 2: DATA PROCESSING REQUIREMENTS

ID	Function Name	Subsystem	Description
FR6	**Real-Time Stream Processing**	IoT Observatory	The system must support data stream processing in real-time (Speed Layer) using Apache Flink and ksqlDB to facilitate immediate analytics.
FR7	**Containerized Algorithm Execution**	Data Hub	The system must utilize Kubernetes to orchestrate the execution of processing algorithms encapsulated in Docker containers, ensuring scalability and resource isolation.
FR8	**Workflow Orchestration (CWL)**	Data Hub	The system must use a workflow controller to execute processing chains defined in CWL, managing scheduling, input staging, and result of publication for EO data.
FR9	**Interactive Analysis Environments**	Data Hub	The system must utilize JupyterHub to dynamically spawn containerized interactive environments (JupyterLab, VS Code, QGIS) for user-driven analysis.
FR10	**On-Demand Tile Rendering**	Data Hub	The system must support dynamic rendering of geospatial data as map tiles (via WMTS) to support interactive visualization without downloading full datasets.

FR11	**Custom Aggregation Execution**	IoT Observatory	The system must allow users to perform aggregations on demand.

2.2.3 DATA MANAGEMENT & STORAGE REQUIREMENTS

This category merges the storage of IoT event streams (Cassandra) with the specialized handling of large geospatial rasters (Object Storage/STAC) required by the Data Hub. Details related to Data Management and Storage Requirements are presented in Table 3.

TABLE 3: DATA MANAGEMENT & STORAGE REQUIREMENTS

ID	Function Name	Subsystem	Description
FR12	**Hierarchical Data Organization**	IoT Observatory	The system must organize IoT data into a strict hierarchy of Organizations, Projects, and Collections to ensure multi-tenancy and data segregation.
FR13	**SpatioTemporal Cataloging (STAC)**	Data Hub	The system must implement the STAC standard to index all EO assets and derived products, decoupling metadata (JSON) from actual data assets.
FR14	**Cloud Optimized Storage (COG)**	Data Hub	The storage layer must support Cloud Optimized GeoTIFF (COG) formats to allow HTTP range requests, enabling the retrieval of specific file segments without full downloads.
FR15	**Persistent User Workspace**	Data Hub	The system must provide persistent storage workspaces (via Object Storage or PVCs) mounted across user sessions to maintain data and tools between logins.
FR16	**Application Repository**	Data Hub	The system must maintain a registry (using Harbor and GitLab) for storing and versioning third-party application packages (Docker images and CWL descriptors).
FR17	**Secure Asset Access**	Data Hub	The system must enforce access control to storage objects via Pre-Signed URLs (STAC Authentication Extension), granting temporary, time-bound access tokens.
FR18	**Data Replication**	Data Hub	The system must maintain data replication (e.g., via Cassandra replication strategies or RAID 6 storage) to ensure fault tolerance and high availability.

2.2.4 APPLICATION GENERATION REQUIREMENTS

Specific to the "Low-Code Platform," these requirements define how the system automates software creation for third parties. Table 4 presents the Application General Requirements.

TABLE 4: APPLICATION GENERATION REQUIREMENTS

ID	Function Name	Subsystem	Description
FR19	**Edge Component Generation**	Low-Code Platform	The platform must allow users to describe edge device characteristics and automatically generate software for embedded devices to handle data dispatch.
FR20	**Logic Component Generation**	Low-Code Platform	The platform must support the automatic generation of data aggregation software and logic flows using Domain-Specific Languages (DSLs).
FR21	**GUI Component Generation**	Low-Code Platform	The platform must allow users to describe visualization requirements to automatically generate the visual parts of the Application GUI components.

2.2.5 DATA ACCESS & VISUALIZATION REQUIREMENTS

This category covers how end-users and external systems interact with the platform outputs. Data access and visualization requirements are presented in Table 5.

TABLE 5: DATA ACCESS & VISUALIZATION REQUIREMENTS

ID	Function Name	Subsystem	Description
FR22	**Complex Querying**	IoT Observatory	The infrastructure must support complex querying capabilities (SQL-like) on raw and processed data streams.
FR23	**Interactive User Environments**	Data Hub	The system must spawn containerized interactive environments (e.g., JupyterLab, QGIS, VS Code) for authenticated users to interact with data.
FR24	**Standardized API Access**	Unified System	The individual parts of the architecture must communicate with external systems through RESTful APIs (OpenAPI v3) and OGC Web Services (WMS/WMTS).
FR25	**Tile-Based Visualization**	Data Hub	The system must provide WMTS endpoints for dynamic rendering of data tiles to support interactive mapping applications.

FR26	**Secure Data Download**	Data Hub	The system must support secure machine-to-machine access to assets via pre-signed URLs to enforce access control on object storage.

2.3 NON-FUNCTIONAL REQUIREMENTS

The non-functional requirements (NFRs) define the quality attributes of the NOSTRADAMUS system, ensuring that the architecture is not only functional but also robust, scalable, secure, and interoperable. These requirements are critical for guiding the selection of the hybrid edge-cloud infrastructure and the specific technologies (e.g., Kubernetes, Kafka, Keycloak) described in this document.

The NFRs are categorized into **five key domains**: 1) Performance & Scalability, 2) Reliability & Availability, 3) Security & Data Protection, 4) Interoperability & Standards, and 5) Maintainability & Operations.

2.3.1 PERFORMANCE AND SCALABILITY REQUIREMENTS

These requirements ensure the system can handle the "High Velocity" and "High Volume" characteristics of the agricultural data streams and Earth Observation archives. Performance and scalability requirements are described in Table 6.

TABLE 6: PERFORMANCE AND SCALABILITY REQUIREMENTS

ID	Requirement	Description	Metric/Target
NFR1	**System Latency**	The IoT Observatory must ensure minimal delay for real-time data ingestion and processing to support immediate decision-making (e.g., frost alerts, water stress, pest detection, etc).	Sub-second latency for real-time streams
NFR2	**Horizontal Scalability**	The system must be able to scale horizontally by adding more nodes to the cluster to accommodate increasing data throughput without downtime.	Linear scalability via Kubernetes
NFR3	**Concurrent User Capacity**	The infrastructure must support simultaneous access and operations by a large number of registered users across the five pilot sites.	> 100 users
NFR4	**High Throughput Ingestion**	IoT Observatory must support concurrent data ingestion from multiple heterogeneous sources (IoT devices, sensors) without performance degradation.	Concurrent ingestion support

2.3.2 RELIABILITY AND AVAILABILITY REQUIREMENTS

These requirements define the system's ability to remain operational and recover from failures, ensuring high veracity and trust. Table 7 presents the reliability and availability requirements.

TABLE 7: RELIABILITY AND AVAILABILITY REQUIREMENTS

ID	Requirement	Description	Metric/Target
NFR5	**System Availability**	IoT Observatory must ensure high uptime to guarantee data access for critical agricultural operations.	99% Availability
NFR6	**Data Replication**	IoT Observatory must maintain data redundancy (e.g., via Cassandra replication or RAID storage) to ensure fault tolerance and complete data recovery in case of hardware failure.	At least 1 backup (Replication Factor > 1)
NFR7	**System Restart Time**	In the event of a service interruption, the IoT Observatory must be capable of a rapid restart cycle to minimize downtime.	< 10 minutes
NFR8	**Disaster Recovery**	If full backup recovery is necessary, the IoT Observatory must restore operations within a strict timeframe.	< 1 hour
NFR9	**Failure Notification**	In case of a serious failure where the IoT Observatory is unable to serve users, it must present a clear notice of temporary unavailability.	User Notification

2.3.3 SECURITY AND DATA PROTECTION REQUIREMENTS

These requirements ensure confidentiality, integrity, and controlled access to data, particularly for sensitive third-party applications and proprietary datasets. Full description of the security and data protection requirements are presented in Table 8.

TABLE 8: SECURITY AND DATA PROTECTION REQUIREMENTS

ID	Requirement	Description	Implementation
NFR10	**Authentication & Authorization**	The system must use state-of-the-practice methods to strictly control access, ensuring users can only access data/projects they are authorized for.	Keycloak (SSO), OAuth2, JWT
NFR11	**API Security**	Access to data APIs must be secured using API keys or tokens, distinguishing between different access levels (e.g., Master, Read, Write keys).	API Tokens

NFR12	**Secure Data Transport**	All data exchanged between users and the system must be encrypted during transmission to prevent interception.	HTTPS / TLS
NFR13	**Data Segregation**	The system must enforce strict isolation between different Organizations and Projects to ensure multi-tenancy privacy.	Keyspaces / Namespaces
NFR14	**Secure Asset Access**	Direct access to stored assets (e.g., in S3) must be time-bound and controlled, preventing unauthorized direct downloads.	Pre-Signed URLs

2.3.4 INTEROPERABILITY AND STANDARDS REQUIREMENTS

These requirements ensure the NOSTRADAMUS architecture functions as a "*System of Systems*" compatible with external EU platforms and third-party tools. Table 9 presents the detailed descriptions of interoperability requirements and relative standards.

TABLE 9: INTEROPERABILITY AND STANDARDS REQUIREMENTS

ID	Requirement	Description	Standard
NFR15	**API Standardization**	The infrastructure must communicate with external systems through standardized RESTful APIs to facilitate third-party integration.	OpenAPI v3
NFR16	**Data Format Compliance**	The system must support standard data exchange formats to ensure compatibility with external tools and dashboards.	JSON / GeoJSON
NFR17	**Geospatial Standards**	The Data Hub must adhere to open geospatial standards for cataloging and visualizing Earth Observation data.	STAC, OGC WMS/WMTS
NFR18	**Workflow Portability**	Processing algorithms must be packaged in a standard, platform-agnostic format to ensure reproducibility across different environments.	Common Workflow Language (CWL)

2.3.5 MAINTAINABILITY AND OPERATIONAL REQUIREMENTS

These requirements support the long-term sustainability and administration of the platform. Maintainability and operational requirements are described in Table 10.

TABLE 10: MAINTAINABILITY AND OPERATIONAL REQUIREMENTS

ID	Requirement	Description
NFR20	**System Logging**	The system must keep detailed activity logs to track usage, diagnose system failures, and support auditing.
NFR21	**Automated Deployment**	The system should support automated deployment and updates of components to reduce manual error and maintenance time.
NFR22	**Modularity**	The architecture must be modular (microservices), allowing individual components (e.g., Data Cubes, IoT Brokers) to be updated or replaced without affecting the whole system.

2.4 VALIDATION AND PRIORITIZATION FRAMEWORK

To ensure that the system requirements defined in this deliverable are robust and aligned with European data strategies, the NOSTRADAMUS project utilizes a Validation and Prioritization Gating[1] framework. Every requirement identified in the needs-gathering phase must pass through three fundamental semantic filters before being operationalized into the architectural design: a) FAIR Principles, b) Actionability, and c) Measurability.

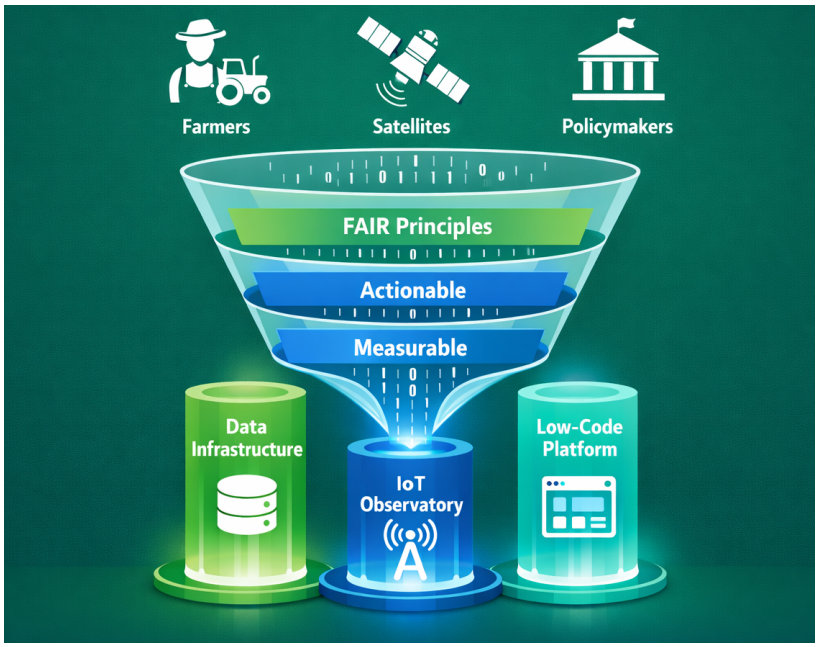

FIGURE 1: VALIDATION AND PRIORITIZATION FRAMEWORK

[1] Hujainah, Fadhl & Bakar, Rohani & Abdulhak, Mansoor & Zamli, Kamal. (2018). Software Requirements Prioritisation: A Systematic Literature Review on Significance, Stakeholders, Techniques and Challenges. IEEE Access. PP. 1-1. 10.1109/ACCESS.2018.2881755.

2.4.1 ADHERENCE TO FAIR PRINCIPLES

The **FAIR**-ification[2] of data and system interactions is a core requirement of the NOSTRADAMUS architecture to ensure long-term sustainability and cross-border utility.

- **Findability**: The system implements the STAC standard and Persistent Identifiers (DOIs) to ensure that all EO assets and IoT datasets are easily discoverable.
- **Accessibility:** Data is made "as open as possible, as closed as necessary" through standardized RESTful interfaces with access control, with secure machine-to-machine access facilitated using API tokens.
- **Interoperability:** By strictly adhering to OGC/ISO standards and utilizing interoperable formats such as Cloud Optimized GeoTIFFs (COGs) and JSON, the platform functions as a "System of Systems" capable of integration with external EU infrastructures like Copernicus.
- **Reusability:** Comprehensive documentation, including a Data Dictionary and rich metadata (YAML), ensures that research outputs can be reproduced and reused by third-party developers in WP6.

2.4.2 ACTIONABLE REQUIREMENTS

The architecture prioritizes requirements that are actionable, meaning they must directly inform a technical implementation or a decision-support outcome.

- **Operationalization**: High-level stakeholder needs are transformed into concrete Functional Requirements (FR1-FR26) that define system behaviors, such as Real-Time Stream Processing or Automated Logic Component Generation.
- **Decision Support:** A requirement is considered actionable if it provides the "Expert Information" necessary for farmers to reduce agrochemical use or for policymakers to assess food security trends.

3 OVERALL SYSTEM ARCHITECTURE

The Nostradamus architecture is conceptualized as a Hybrid Edge-to-Cloud "System of Systems" specifically engineered to tackle the significant data fragmentation challenges prevalent across European agriculture sector. This innovative platform is meticulously aligned with crucial European frameworks, including the European Green Deal, the Common Agricultural Policy (CAP), and the broader European strategy for data, thereby establishing a robust, data-centric foundation to bolster the EU's food security and self-sufficiency.

To achieve its goals, the Nostradamus ecosystem employs a "separation of concerns" strategy, segmenting its cloud-native infrastructure into distinct, yet highly integrated, functional pillars. This

[2] https://www.go-fair.org/fair-principles/

architecture not only bridges the gap between initial agricultural requirements and the practical development of prototypes, data pipelines, and end-user applications, but it also critically promotes a low-code model-driven approach for application development. This particular innovation is central to the project's vision, as it empowers IT experts, such as SMEs, Agri-industry, Cooperatives, Agri-food companies, or even scientists to construct customized digital solutions without programming expertise, thereby alleviating the traditional technical burden and accelerating the creation of relevant, real-world agricultural applications.

3.1 VISION AND PRINCIPLES

The NOSTRADAMUS architecture is designed as a Hybrid Edge-to-Cloud "System of Systems" specifically engineered to solve the data fragmentation challenges currently facing European agriculture. In strict alignment with the European Green Deal, the Common Agricultural Policy (CAP), and the European strategy for data, the platform establishes a robust, data-centric foundation to support the EU's food security and self-sufficiency. The overall system is architected to conquer the "**four V's of Big Data**", managing the massive **Volume** of Earth Observation (EO) archives, the high **Variety** of heterogeneous IoT and socio-economic data, the sub-second **Velocity** required for near real-time decision-making, and the high **Veracity** necessary for reliable and accurate decision support

The architectural vision is built upon the following core principles:

A) **Modularity and Microservices:** The system is composed of loosely coupled components (Data Cubes, IoT Brokers, Processing Engines) deployed as containerized microservices. This ensures that individual modules can be updated, scaled, or replaced without disrupting the broader ecosystem.

B) **Open Standards and Interoperability:** To prevent vendor lock-in and ensure wide adoption, the architecture strictly adheres to open standards, including OGC (Open Geospatial Consortium) standards for geospatial data, STAC for data discovery, and OpenAPI v3 for interface specifications.

C) **Lambda and Kappa Architecture Fusion:** To reconcile the conflicting needs of real-time monitoring (low latency) and historical analysis (high accuracy), the data processing layer implements a hybrid Lambda/Kappa architecture. This allows the system to process high-velocity streams in real-time while simultaneously maintaining a comprehensive, immutable master dataset for batch processing.

D) **Cyber-Physical Integration:** The architecture treats the agricultural environment as a Cyber-Physical System (CPS), where digital applications not only monitor physical assets (sensors, crops) but can also trigger actions (actuators) through a seamless Edge-to-Cloud continuum.

3.2 ACTOR INTERACTIONS WITH THE DIGITAL ECOSYSTEM

To ensure the NOSTRADAMUS System-of-Systems approach remains demand-driven and scientifically robust, the platform utilizes a Multi-Actor Approach (MAA). The operationalization of the architecture distinguishes between two primary archetypes based on their technical interaction with the Low-Code Platform (III). This categorization ensures that the deep technical burden of cyber-physical systems remains with technical intermediaries, while domain stakeholders are empowered to steer the development process through high-level abstractions.

In summary, the mapping of Stakeholder Archetypes to their specific roles in the architecture is defined below:

- **IT Providers** (TG1, TG7, TG2)
- **Domain Experts** (TG3, TG5, TG6, TG2)

For convenient, the TG Mapping as defined in the GA is listed below:

- TG1: IT solution providers: SMEs, entrepreneurs, and software developers who utilize the project's open-source tools to build applications.
- TG2: Research & Academia: Universities and institutes engaging in cross-border research and digitalization of the agricultural sector.
- TG3: Policy Makers: Regional, national, and EU authorities using open-source solutions for better data collection and policy evaluation.
- TG4: Investment advisors in agriculture: Manufacturers of agrichemicals, seeds, and machinery looking for cutting-edge digital insights.
- TG5: Land Managers: Farmers and foresters who benefit from production efficiency and reduced costs through the platform's outputs.
- TG6: Agricultural advisory services: Extension services and consultants who use NOSTRADAMUS solutions to provide better advice to farmers.
- TG7: Ag-tech industry: Precision agriculture and robotic companies developing new products based on the project's results.
- TG8: Society: Consumers, NGOs, and citizens interested in the digitalization and sustainability of the agri-food sector.

3.2.1 TECHNICAL INTERMEDIARIES AND IT PROVIDERS

This group constitutes the primary "builders" of the digital applications that populate the NOSTRADAMUS ecosystem (TG1, TG7).

- **Target Groups:** Small and Medium-Sized Enterprises (SMEs), agro-industry providers, tech startups, and spin-offs in software development.
- **Role in Architecture:** These actors are the primary participants in the WP6 Open Calls. They utilize the platform to develop, test, and validate interoperable digital applications covering the full Edge-to-Cloud continuum.

- **Interaction Model:** IT providers interact directly with the Application Generation Engine (AGE) and the Domain-Specific Languages (DSLs). They leverage the bootstrapping software and Model-Driven Engineering (MDE) artifacts to rapidly synthesize complex logic flows and connectivity code.

3.2.2 DOMAIN EXPERTS AND CITIZEN DEVELOPERS

This group represents the essential agricultural stakeholders whose domain knowledge defines the system's requirements and whose operations benefit from its outputs (TG3, TG5, TG6).

- **Target Groups:** Farmers, land managers, agronomists, agricultural advisors, and policymakers.
- **Role in Architecture:** These actors are primarily the steerers and end-users of the NOSTRADAMUS-powered applications. They utilize the generated dashboards and core modules for real-world farm decision-making and policy evaluation.
- **Interaction Model:** To lower the technical entry barrier, the platform promotes a "citizen developer" concept. Through the AI Assistant, these actors participate in the generation of DSL models by providing natural language descriptions of their monitoring workflows. This "Vibe Coding" paradigm allows them to design solutions based on "what" they need rather than "how" to program them.

3.3 HIGH-LEVEL ARCHITECTURE OVERVIEW

By bridging the gap between field-level event acquisition and high-level cloud-native analytics, the system provides a consolidated, scalable, and interoperable environment for agricultural decision support. To achieve this architectural vision, the platform implements a separation of concerns strategy, dividing the cloud-native infrastructure into three distinct but highly integrated functional pillars: **Data Infrastructure (I)**, the **IoT Observatory (II)**, and the **Low-Code Platform (III)**. As illustrated in Figure 2, these layers work in concert to ingest raw data, transform it into actionable intelligence, and deliver it to end-users through automated application generation.

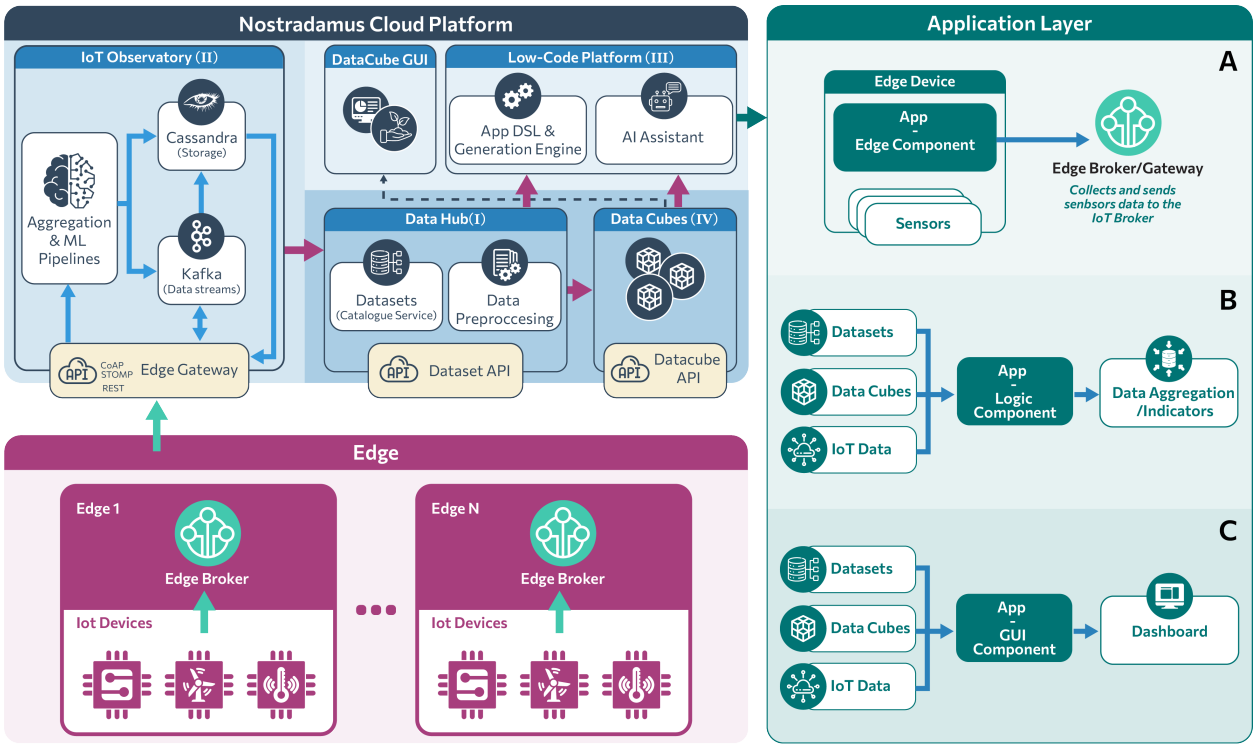

FIGURE 2: NOSTRADAMUS OVERALL ARCHITECTURE

The architectural framework is divided into three distinct but highly integrated logical layers that work in concert to ingest raw data, transform it into actionable intelligence, and deliver it to end-users via a low/no-code engine that abstracts technicalities and focus on the design and the automated generation of NOSTRADAMUS applications. The design conforms to the separation of concerns design pattern, as evident in Figure 2, providing 3 individual but interoperable Layers:

A) **The Data Infrastructure Layer:** This layer serves as the "Batch Layer" and long-term storage engine. It hosts the Data Hub and Data Cubes, responsible for managing massive static datasets, Earth Observation (EO) archives, and socio-economic indicators. It provides means for multi-dimensional arrays that harmonize geospatial data spatially and temporally, to provide a structured way to store and process information from missions such as Sentinel-1/2 and Landsat.

B) **The IoT Observatory Layer:** This layer serves as the "Speed Layer." It is a specialized Big Data Management System (BDMS) designed to ingest, buffer, and process high-velocity sensor data streams with sub-second latency. It is engineered for real-time event handling, by employing a stack of distributed technologies, including Apache Kafka for persistent buffering and Apache Flink for complex event processing, to ingest and process high-throughput sensor streams with sub-second latency.

C) **The Low-Code Application Layer:** This layer acts as the interface between the technical infrastructure and the domain experts. Acting as the democratization layer, this component provides the tools necessary for domain experts (SMEs, Agri-industry, Cooperatives, Agri-food companies) to build custom digital solutions. By leveraging Model-Driven Engineering

and Domain-Specific Languages (DSLs), the platform utilizes an Application Generation Engine to automatically transform description models into executable software artifacts, alleviating the technical burden of direct programming. Farmers, agronomists, and policy makers are the end-users of the applications powered by NOSTRADAMUS. According to the grant agreement, such applications will support professionals for 2 main activities:

a. Digital applications for farm decision-making (notably based on processing/rendering of farming business data)
b. Digital applications for policy making and evaluation (notably based on processing/rendering of socio-economic data on farmers and rural communities)

The architectural integrity of the system is built upon the fusion of Lambda and Kappa patterns, a hybrid approach that ensures the platform can calculate real-time ad-hoc user queries while simultaneously maintaining an immutable "source of truth" master dataset for comprehensive historical analysis. Furthermore, the system adopts an API-centric approach, exposing all core functionalities via standardized REST endpoints (OpenAPI v3) and OGC Web Services (WMS/WMTS). This ensures that the NOSTRADAMUS ecosystem remains scalable, modular, and fully interoperable with external EU platforms and third-party tools.

Ultimately, by bridging the gap between field-level IoT data acquisition and high-level cloud-native analytics, this architecture treats the agricultural environment as a Cyber-Physical System (CPS). This creates a seamless Edge-to-Cloud continuum where digital applications not only monitor physical assets like crops and machinery but can also trigger actionable decisions and automated responses to improve yields and reduce agrochemical reliance.

3.3.1 DATA INFRASTRUCTURE

The Data Infrastructure functions as the system's "Batch Layer" and long-term storage engine, responsible for managing massive static datasets and Earth Observation (EO) archives. It provides the foundation for "hard" data handling, including satellite imagery from missions such as Sentinel-1/2 and Landsat.

- **Open Data Cubes:** At the core of this layer is an ensemble of Data Cubes, multi-dimensional arrays that harmonize geospatial data spatially and temporally. This abstraction allows researchers to perform time-series analysis and "pixel-drilling" without the need to manually manage thousands of individual satellite scenes.
- **SpatioTemporal Asset Catalog:** To ensure high performance in data discovery, the infrastructure implements the STAC standard. This decouples metadata (JSON) from the actual data assets (COGs), allowing applications to perform complex spatiotemporal queries—such as filtering by cloud cover or specific geographic polygons—before any heavy data transfer occurs.
- **Data Preprocessing and Pipelines:** The layer incorporates automated pipelines for converting raw EO data into Analysis Ready Data (ARD). These pipelines utilize Common Workflow Language (CWL) and Docker containers to ensure that data acquisition,

atmospheric correction, and harmonization processes are fully reproducible and portable across different cloud environments.

- **Cloud Optimized GeoTIFFs:** By storing raster data in COG format, the infrastructure enables HTTP range requests. This allows applications to retrieve only the specific byte-ranges required for a visualization or analysis task, drastically reducing latency and computational overhead.

The Data Infrastructure of NOSTRADAMUS is implemented through two complementary subsystems with clearly defined responsibilities. The EO Data Hub is responsible for the ingestion, cataloguing, processing, and distribution of EO data. It provides a cloud-native environment supporting scalable data pipelines, metadata management based on the STAC specification, and execution of processing workflows using containerized applications described with the CWL.

The Data Hub ensures that EO data are transformed into ARD and made available through standardized interfaces. These data products are then consumed by the Data Cube infrastructure, developed by UNIGE, which focuses on structuring, indexing, and enabling analytical access to multi-dimensional datasets. This separation of concerns allows the Data Hub to act as the data provisioning layer, while the Data Cubes provide the analytical abstraction layer.

The interaction between these components is based on standardized metadata and interoperable data formats, ensuring seamless integration between data ingestion, processing, and higher-level analytics. This architecture enables scalability, reproducibility, and flexibility in supporting both batch and on-demand data processing workflows across the platform.

3.3.2 IOT OBSERVATORY

The IoT Observatory (IoTO) is a central and critical component of the NOSTRADAMUS Cloud Platform, designed to address the significant challenges associated with real-time and historical data in agricultural contexts. It functions as a specialized Big Data Management System (BDMS) engineered for real-time event handling, ingesting and processing high-velocity sensor data streams with sub-second latency. The primary objective of the IoTO is to transform raw agricultural sensor data into actionable intelligence, thereby supporting informed decision-making across the agricultural sector.

The fundamental unit of data managed by the IoTO is "events," such as sensor readings and actuator states, which are collected from IoT devices. This infrastructure is not merely a database but a complex orchestration of distributed systems focused on effective data management, real-time monitoring, and advanced analytics. The design prioritizes minimal latency for real-time monitoring while also accommodating complex, long-term historical analysis.

To reconcile the inherent tension between the need for immediate, real-time insights and comprehensive historical accuracy, the IoT Observatory employs a sophisticated hybrid

architectural pattern that fuses elements from both Lambda and Kappa architectures. This hybrid approach offers several key advantages:

- Robustness against data latency issues: By separating real-time and batch processing, the system prevents processing bottlenecks even during peak data.
- Capability to replay and reprocess historical data streams: This is crucial for refining or updating algorithms, ensuring that analyses remain current and accurate over time.

The IoT Observatory functions as one of the three primary pillars of the overall NOSTRADAMUS hybrid Edge-to-Cloud architecture (Pillar II in Figure 2). It acts as the gateway for edge and IoT devices to connect, send, and aggregate data via a unified API. It provides an abstraction layer for connecting to domain-independent Edge systems through an IoT broker, known as the Edge Gateway, which supports communication protocols like MQTT and CoAP. Its outputs are crucial for informed decision-making support and for empowering the development of open-source digital applications by various third parties within the ecosystem.

For comprehensive information and detailed technical specifications of the IoT Observatory, including its technology stack, data schema, security considerations, data flows, and API endpoints, refer to Section 4.2.

3.3.3 LOW-CODE APPLICATION PLATFORM

The Low-Code Application Platform (LCP) is a pivotal and distinguishing component of the NOSTRADAMUS Cloud Platform, specifically designed to address the challenges of complex cyber-physical application development within the agricultural sector. Its primary objective is to democratize the creation of digital solutions, allowing domain experts such as SMEs, Agri-industry, Cooperatives, and Agri-food companies to build custom applications without requiring technical knowledge. This approach aims to enhance resource efficiency, reduce reliance on agrochemicals, and strengthen resilience against potential disruptions, aligning with key EU frameworks like the European Green Deal and the Common Agricultural Policy.

The LCP's core functionality revolves around enabling the rapid deployment of data-driven digital solutions through abstraction and automation. The platform employs a Model-Driven Engineering (MDE) approach, utilizing **Domain-Specific Languages (DSLs)** to allow experts to design, generate, and deploy software for different parts of their applications. This MDE paradigm promotes a "citizen developer" concept, where non-technical users can auto-generate parts or the entirety of their applications simply by designing the solution rather than writing code ("what" instead of "how"). This process is intended to make real-world application development easy, fast, and error-free. DSLs are specialized languages that allow experts to describe specific aspects, such as edge device characteristics, sensor configurations, visualization requirements, and logical data flows, without needing general-purpose programming expertise.

Another key innovation within the NOSTRADAMUS LCP is its **AI assistant**, which is supported by a multi-agent backend. This AI assistant acts as a natural language interface, allowing domain experts to describe their desired monitoring workflows using plain text. The AI then translates these natural language descriptions into formal DSL models, which are subsequently fed into the Application Generation Engine. This mechanism significantly reduces the technical burden and latency traditionally associated with programming complex cyber-physical systems.

The LCP's development is a specific deliverable of Work Package 2 (WP2). The Aristotle University of Thessaloniki (AUTH) leads Task 2.4, which is "*Develop generation engine for digital applications*". This architecture serves as the foundation for Deliverable D2.5 ("Application Generation Engine") and its subsequent version, D2.6. It defines the structural rules, component interactions, and "Edge-to-Cloud" logic that the Application Generation Engine will automate. Its outputs are integral to Work Package 6 (WP6), which focuses on third-party funding for digital applications and capacity building, ensuring that applications developed by funded third parties are interoperable with the NOSTRADAMUS ecosystem. AUTH also contributes to training IT experts (SMEs, Agri-industry, Cooperatives, Agri-food companies) on how to use the LCP to develop open-source digital applications.

For comprehensive information and detailed technical specifications of the Low-Code Application Platform, please refer to Section 4.4.

3.4 HYBRID APPLICATIONS

The digital solutions developed within the NOSTRADAMUS ecosystem are defined as cyber-physical applications that orchestrate a resource continuum spanning the agricultural field, the network edge, and the cloud. By bridging the gap between field-level sensory events and high-level analytical insights, these applications treat the agricultural environment as an integrated Cyber-Physical System (CPS).

To democratize development and alleviate the deep technical burden traditionally associated with programming such hybrid systems, the architecture decomposes applications into three distinct types of interoperable components:

- **Edge Components (A):** Facilitating primary sensory acquisition and local actuation at the physical interface.
- **Logic Components (B):** Serving as the cloud-based analytical engine for data fusion and indicator generation.
- **GUI Components (C):** Constituting the user-facing layer for visualization and interactive decision support.

Edge Components are cyber-physical in nature, as these are applications deployed at the edge, providing means of observation from installed sensors in the field, and data feed to the NOSTRADAMUS Cloud via the Edge Gateway of the IoT Observatory. On the other hand, Logic and GUI

Components are considered completely digital; these components use data from the Data Hub, IoTO and Data Cubes and produce higher level of information, such as indicators and visualizations.

As illustrated in Figure 2, these tiers work in concert to transform raw data into actionable intelligence through a structured, quality-aware process. While this section establishes the high-level conceptual framework, a comprehensive technical breakdown of these application tiers, including their specific data flows, connectivity patterns, and the low-code generation workflow, is detailed in Section 6: Cloud, Edge, and Mixed Applications.

3.5 HYBRID CONNECTIVITY AND SECURITY MODEL

The integrity of the NOSTRADAMUS "System of Systems" model is maintained through a unified connectivity and security framework that spans the cloud and edge environments. This framework is defined by the below characteristics.

- **API-Centric Integration:** All functionality is exposed via standardized REST endpoints (OpenAPI v3) and OGC Web Services (WMS/WMTS), ensuring that external tools and third-party developers can seamlessly integrate with the platform's core services
- **Hybrid Deployment Models:** The architecture supports flexible deployment schemas where processing is distributed between the Cloud Layer (for heavy lifting and historical truth) and the Edge Layer (for immediate, low-latency reaction and localized data filtering)
- **Security and Access Control:** Access management is centralized via Keycloak, supporting OAuth2 and OpenID Connect. Security is enforced at two levels: human users are authenticated via JWT tokens (SSO), while machine-to-machine communication is secured through a tiered API Key system (Master, Read, and Write keys) to strictly isolate project data.
- **Secure Asset Access:** Direct access to massive raster assets in object storage (S3) is managed through pre-signed URLs. These time-bound tokens grant temporary, authorized download permissions, ensuring data sovereignty and preventing unauthorized access to the storage backend.

4 ARCHITECTURAL PILLARS

To achieve the scalability and modularity required for a continental-scale agricultural platform, the system is decomposed into specialized modules that function as loosely coupled microservices. This section provides a detailed technical breakdown of the core components, including the Data Hub for Earth Observation (EO) archives, the event-driven IoT Observatory for real-time telemetry, and the Low-Code Platform for application orchestration. Each component is integrated through standardized OpenAPI v3 and OGC interfaces, allowing researchers and developers to interact with the platform's analytical capabilities through a unified service model.

4.1 DATA HUB

The functional architecture of the NOSTRADAMUS Data Hub is composed of modular services that support data ingestion, processing, discovery, access, and interactive user workflows. These services are deployed in a container-based infrastructure and integrated through standardized APIs. The system is designed to support multiple roles and access patterns, enabling data scientists, EO specialists, and developers to interact with datasets, develop algorithms, and execute processing workflows within the same operational environment.

Within the NOSTRADAMUS architecture, the Data Hub provides the operational EO data service layer. Its scope covers EO data ingestion, metadata extraction and cataloguing, data access services, and the execution of processing workflows. The Data Hub acts as the main provisioning environment for EO datasets and derived products that are consumed by downstream components and user-facing applications.

The Data Hub is functionally distinct from the Data Cube infrastructure developed by UNIGE. While the Data Hub focuses on the acquisition, preparation, and processing of EO data, including the generation of Analysis Ready Data, the Data Cube infrastructure focuses on the structuring, indexing, and analytical exploitation of harmonized multidimensional datasets. This separation ensures a clear interface between data provisioning and data analytics while maintaining interoperability through shared standards and APIs.

The implementation supports several WP3 activities. In Task 3.4, it provides the core services for EO data ingestion and provision; in Task 3.5, it enables the integration and execution of processing algorithms through containerized workflows; and in Task 3.6, it is deployed and operated within the project cloud infrastructure. This positions the Data Hub as a central component connecting EO data sources, processing services, and higher-level NOSTRADAMUS modules.

4.1.1 SYSTEM ARCHITECTURE

The internal architecture of the Data Hub (I), illustrated in Figure 3, is designed as a modular, cloud-native platform that supports the complete lifecycle of EO data, from automated ingestion to interactive scientific analysis. The system is organized into loosely coupled service layers that ensure scalability, reproducibility, and interoperability using open-source technologies, protocols and standards.

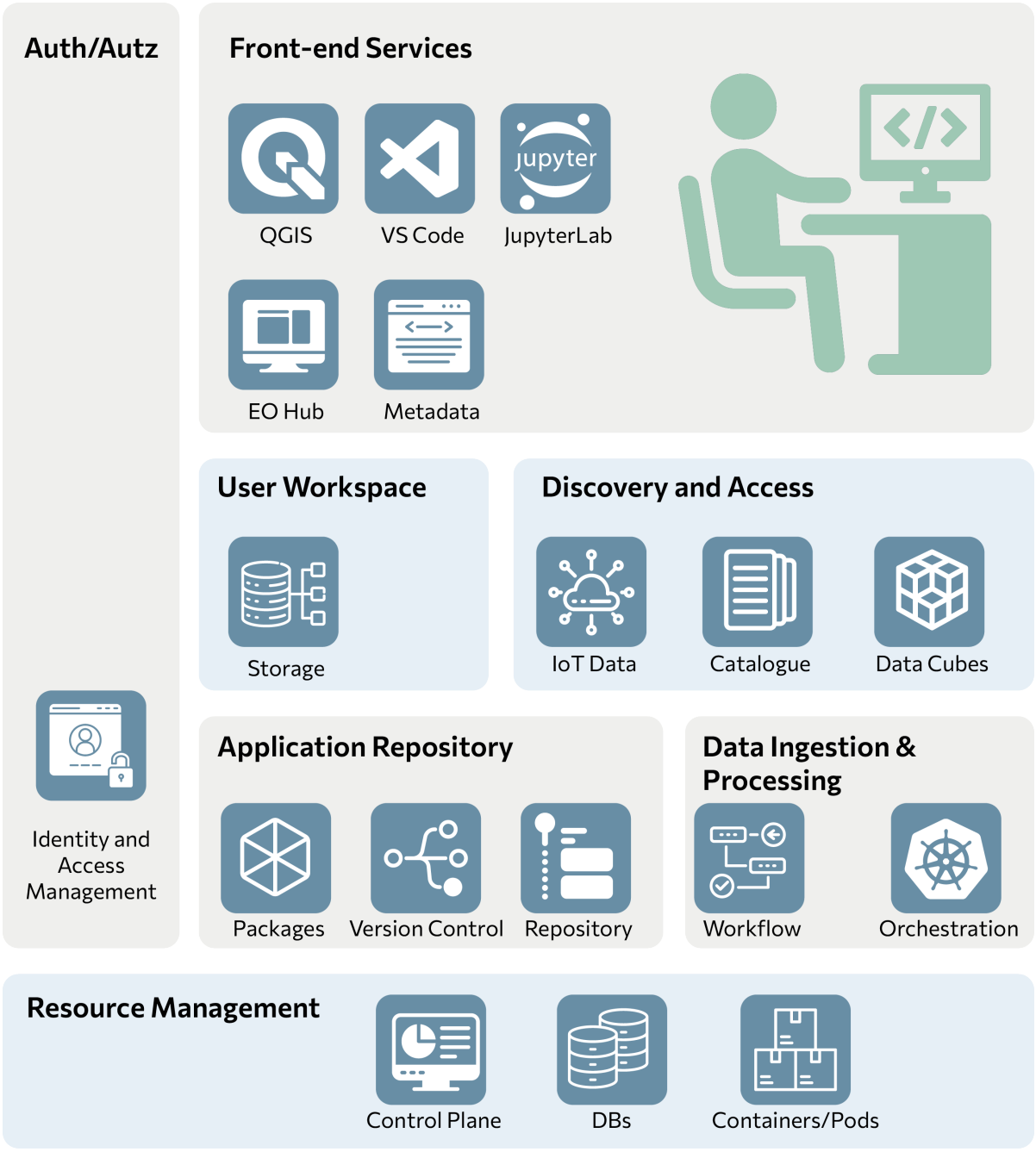

FIGURE 3: DATA HUB INTERNAL ARCHITECTURE

The functional components depicted in the architecture of Data Hub are categorized as follows:

- **Front-end Services:** This layer serves as the primary entry point for researchers and data scientists. It includes the EO Hub, a web-based portal for discovering services and data collections and utilizes JupyterHub to dynamically spawn containerized interactive

environments such as JupyterLab, VS Code, and QGIS. These environments are isolated and pre-configured with necessary EO toolboxes (e.g., GDAL[3], SNAP[4]).

- **User Workspace:** To facilitate iterative development, the architecture provides a persistent storage area (S3-compatible) mounted across different user sessions. This allows users to maintain notebooks, intermediate results, and custom tools in a secure, project-scoped environment.
- **Discovery and Access:** This core layer handles the indexing and provisioning of data assets. It utilizes the STAC standard to decouple metadata from binary files, enabling granular, real-time searches via the Catalogue API. It also provides the interface for Data Cubes (IV), which offer harmonized, multi-dimensional gridded arrays for high-performance spatiotemporal analysis.
- **Algorithms Repository:** This component acts as a centralized registry for processing algorithms. Algorithms are packaged using the Common Workflow Language (CWL), version-controlled via GitLab, and stored as Docker images in a Harbor repository. This ensures that all analytical tools are portable and reproducible across different cloud environments.
- **Data Ingestion & Processing:** This layer orchestrates the execution of computational pipelines. It utilizes a Workflow Controller to manage CWL-based sequences, such as data acquisition, atmospheric correction, and tiling, which are executed as containerized pods within a Kubernetes orchestration environment.
- **Identity and Access Management (Auth/Autz):** Security is managed through a unified layer powered by Keycloak, supporting industry standards such as OAuth2 and OpenID Connect. This ensures centralized authentication and role-based access control across all hub services.
- **Resource Management:** The underlying infrastructure is managed by a Control Plane that handles the scheduling and monitoring of compute and storage resources. This layer provides the necessary observability to maintain high availability and reliability under dynamic workloads.

By integrating these specialized layers, the Data Hub provides a cohesive environment that transforms raw satellite imagery and in-situ data into ARD, directly supporting the project's core agricultural modules and third-party digital applications.

4.1.2 DATA DISCOVERY AND ACCESS

The platform includes a discovery and access layer that exposes EO datasets through a STAC compliant catalogue service. This service allows users to search, filter, and preview data based on spatial and temporal criteria. Each dataset is described through structured metadata, and its associated assets can be accessed securely via signed URLs or API-based downloads.

[3] https://gdal.org/en/stable/index.html
[4] https://step.esa.int/main/toolboxes/snap/

Beyond catalogue access, the platform supports data cube abstractions that enable interaction with pre-structured, gridded EO datasets. These data cubes facilitate time-series analytics, consistent tiling, and spatial harmonization. They are especially useful in scientific or operational workflows where reproducibility and performance are key. Integration with the processing environment allows users to access data cubes both in notebook sessions and through CWL pipelines

4.1.3 SOFTWARE PACKAGING AND INTEGRATION

The NOSTRADAMUS platform adopts a standardized approach for packaging and integrating data processing software, enabling reproducible, portable, and scalable execution of workflows across the infrastructure. This approach is implemented within the Data Hub and is aligned with best practices for EO processing services.

Software can be packaged as containerized processing workflows, ensuring that all dependencies, libraries, and runtime environments are encapsulated within the execution unit. This enables consistent deployment across different computing environments and supports the portability of processing chains.

The processing workflows are described using the CWL, which provides a standardized and declarative way to define data processing steps, inputs, outputs, and execution logic. This allows complex EO processing chains to be composed, shared, and reused across the platform.

The integration of software packaging into the platform follows a structured process, including validation of input/output specifications, registration of metadata, and deployment within the orchestration environment. Once integrated, this type of software can be executed on-demand or as part of automated pipelines, leveraging the underlying cloud infrastructure for scalable processing.

This approach supports the onboarding of modules developed within WP4 (e.g., domain-specific analytical modules) and ensures their seamless integration with the Data Hub, Data Cubes, and other platform components. It also enables interoperability with external processing environments using standard interfaces and container-based execution models. Packaged software is deployed and executed within the Data Hub environment, leveraging container orchestration and workflow management services to support scalable and reproducible processing.

4.1.4 DATA INGESTION AND PROCESSING

Data ingestion and processing are implemented as declarative workflows written in CWL and executed within the container orchestration environment. Ingestion pipelines handle steps such as data acquisition, conversion, metadata extraction, and registration into the catalogue. They may also include pre-processing operations that align data with the data cube structure, such as tiling, reprojection, or format harmonization.

Processing workflows follow a similar pattern. Each step is described through Common Workflow Language (CWL) specifications, and the full chain is executed through a workflow controller that manages scheduling, input staging, and results of publication. This enables automated, reproducible pipelines that integrate seamlessly with the user's workspace and application repository.

The orchestration layer is based on Kubernetes-native services and supports scaling, monitoring and recovery. Workflow execution can be triggered manually by users or automatically as part of scheduled or event-based pipelines, depending on operational needs.

The detailed design and execution of EO data ingestion and processing pipelines are described in Section 5, where the implementation of these workflows within the Data Hub environment is further elaborated.

4.1.5 OPERATIONS

Operational services are deployed and monitored through a control plane that manages the lifecycle of platform components. This includes workload scheduling, service scaling, fault recovery, and resource allocation. The system uses Kubernetes for container orchestration, enabling policy enforcement and resource balancing across namespaces and user sessions. Workflow execution and event-driven processing are supported through orchestration frameworks such as Argo Workflows and Argo Events, enabling the automation and coordination of processing pipelines and operational tasks.

Authentication and authorization are managed through a centralized identity provider based on OAuth2 and OpenID Connect standards (e.g. using Keycloak), issuing tokens for access to both interactive environments and programmatic APIs. All service interactions are logged and can be audited, and resource consumption is tracked through integrated observability dashboards. rom an operational perspective, this layer is responsible for the deployment, scaling, monitoring, and lifecycle management of platform services, while application-specific logic and data processing pipelines are handled in dedicated components described in subsequent sections.

Monitoring and observability are implemented using a combination of industry-standard tools, including Prometheus for metrics collection and alerting, Grafana for visualization and operational dashboards, and centralized logging solutions (e.g. ELK stack) for log management and analysis. These tools provide visibility into workflow performance, storage usage, and application health, enabling administrators to maintain the platform reliability and performance under dynamic workloads.

4.2 IOT OBSERVATORY

The IoT Observatory represents the central and critical system of the NOSTRADAMUS platform, designed specifically to address the challenges of "Big Data" in agriculture. It functions as an

Integrated Big Data Management System (BDMS) capable of ingesting, processing, storing, and serving data characterized by the **"Four Vs"**: *High Volume, High Variety, High Velocity, and High Veracity*.

This component is not merely a database, but a complex orchestration of distributed systems designed to handle the fundamental unit of agricultural data—events (e.g., sensor readings, actuator states). The design prioritizes sub-second latency for real-time monitoring while simultaneously supporting complex, long-term historical analysis.

4.2.1 SYSTEM ARCHITECTURE

To reconcile the conflicting requirements of real-time speed and historical accuracy, the IoT Observatory architecture implements a hybrid fusion of the Lambda and Kappa architectural patterns. This hybrid approach ensures the system remains robust against data latency issues while allowing for the reprocessing of historical data streams when algorithms are updated.

The architecture is strictly divided into three functional layers, as evident in Figure 4:

1. **The Batch Layer (Master Dataset):** This layer is responsible for managing the "master dataset"—an immutable, append-only set of raw data. It handles high-latency, complex batch processing to provide comprehensive views of data over time. Crucially, the Batch Layer serves as the "source of truth," correcting any potential inaccuracies or data gaps that may occur in the real-time speed layer.
2. **The Speed Layer (Real-Time Views):** Designed to minimize latency, this layer processes data streams immediately upon ingestion. It compensates for the high latency of the Batch Layer by providing real-time views of the most recent data. This layer utilizes stream processing technologies (Apache Flink, ksqlDB) to update views continuously as new events arrive.
3. **The Serving Layer:** This layer acts as the unified interface for end-users and applications. It merges the real-time views from the Speed Layer with the batch views from the Batch Layer to provide a seamless query experience. It creates an abstraction where the user does not need to know if the data is coming from memory (speed) or disk (batch)

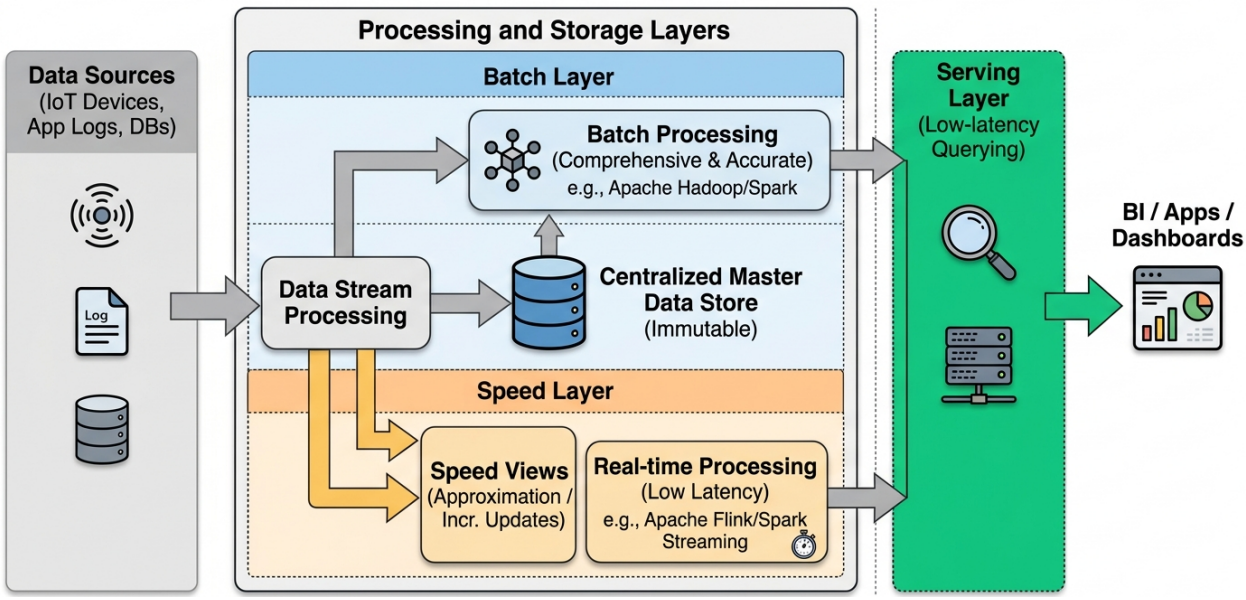

FIGURE 4: LAYERED ARCHITECTURE OF THE NOSTRADAMUS IOT OBSERVATORY

To skillfully navigate the inherent tension between the need for immediate, real-time speed and the demand for comprehensive historical accuracy, the IoTO adopts a sophisticated hybrid architectural pattern. This design strategically fuses key elements from both the Lambda and Kappa architectures. This hybrid methodology offers several critical advantages; it inherently ensures system robustness against data latency issues, preventing processing bottlenecks even during peak data influx. Furthermore, it provides the crucial capability to replay and reprocess historical data streams, which is invaluable when algorithms are refined or updated, thus ensuring that analyses remain current and accurate over time. The entire architecture is logically structured into three distinct yet highly integrated functional layers: a) the Batch Layer (responsible for the Master Dataset and historical processing), the Speed Layer (dedicated to generating Real-Time Views), and the Serving Layer. The Serving Layer acts as a unified interface for end-users and applications, seamlessly merging real-time insights with historical data to provide a cohesive and low-latency query experience, abstracting the data's origin from the user.

4.2.2 TECHNOLOGY STACK AND COMPONENT DESCRIPTION

The IoT Observatory is built upon a stack of open-source, industry-standard technologies chosen for their horizontal scalability and fault tolerance. The technology stack facilitates the management of agricultural data, where the fundamental unit is events, such as sensor readings. The system is designed for interoperability and extensibility, offering two primary high-level APIs for interaction; a REST API and an MQTT/CoAP API.

The technology stack underpinning the IoTO is intentionally built upon a foundation of open-source, industry-standard technologies. This deliberate choice is driven by the imperative for horizontal scalability, allowing the system to expand effortlessly by adding more nodes as data throughput

increases, and for inherent fault tolerance, ensuring continuous operation even in the face of component failures. This technological philosophy aligns with the broader NOSTRADAMUS vision of a modular, microservices-based system that rigorously adheres to open standards, thereby preventing vendor lock-in and fostering widespread adoption and interoperability.

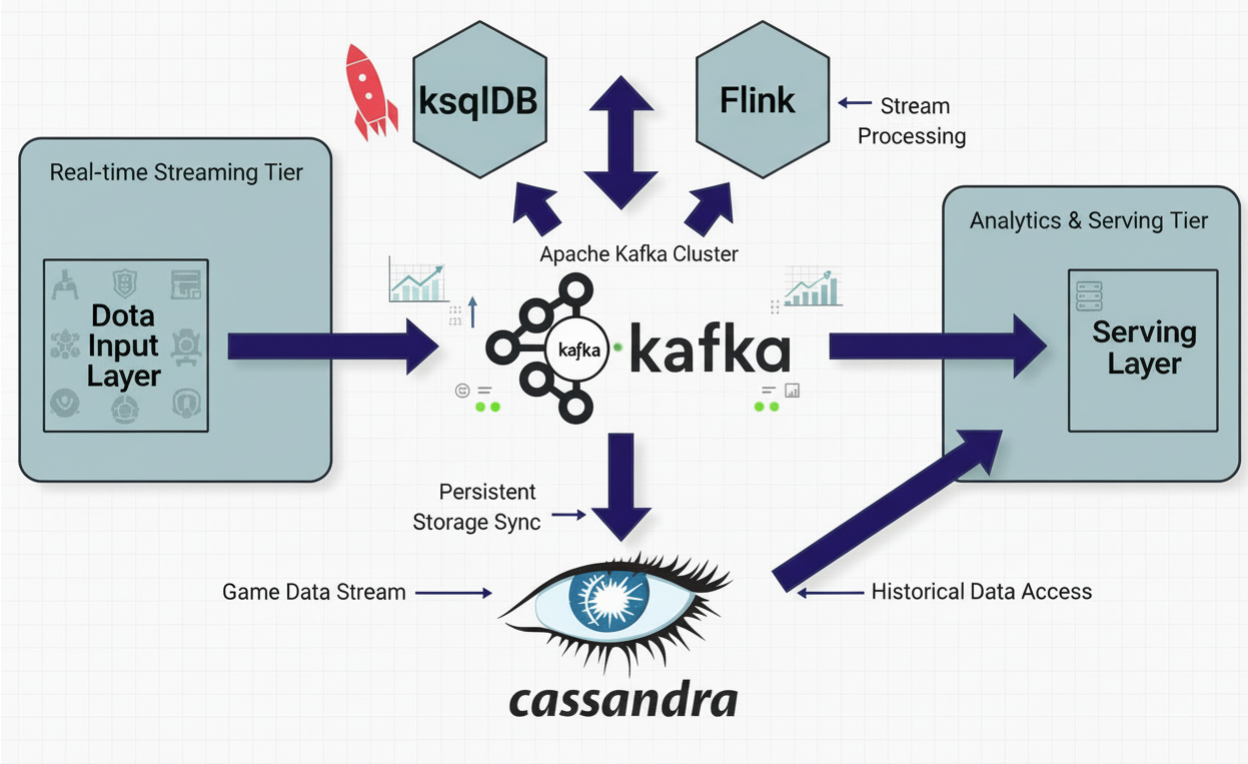

FIGURE 5: IOT OBSERVATORY TECHNOLOGY STACK AND INTEGRATION

Functionally, the IoTO serves as one of the three primary pillars (Pillar II) within the overall NOSTRADAMUS hybrid Edge-to-Cloud architecture. Its role is pivotal in transforming raw, agricultural sensor data into actionable intelligence, which is critical for informed decision-making support across the agricultural sector. Furthermore, it provides the essential backbone for the development of open-source digital applications by various third parties, empowering innovation within the ecosystem.

Data Input Layer: The Edge Gateway

The entry point for all external data is the **Edge Gateway**, a robust abstraction layer designed to handle the chaotic nature of agricultural connectivity. It ensures interoperability with diverse IoT hardware by exposing two primary high-level interfaces:

- **REST API:** For standard, request-response data submission from capable devices and third-party cloud bridges.

- **MQTT:** Optimized for communication from the Edge.
- **CoAP (Under Development):** optimized for lightweight, low-bandwidth communication, essential for constrained IoT devices operating in remote fields with poor connectivity

This layer is engineered for sub-second latency in real-time data ingestion and processing (NFR1), enabling immediate decision-making capabilities, such as frost alerts. Acting as an Apache Kafka producer, the Edge Gateway publishes the ingested data on a specific Kafka topic. This crucial step decouples data producers from consumers, buffering the data to prevent processing logic from being overwhelmed by sudden spikes in data ingestion, such as those caused by extreme weather events. The IoT brokers, including the Edge Gateway, handle the data flow from and to edge devices, contributing a "live" data component to the system.

The Edge Gateway forms the cloud-side interface for App-Edge Components. These software modules are executed on embedded devices (e.g., Raspberry Pi, ESP32 boards) in the field and are responsible for acquiring sensory data and executing local actuation logic. The App-Edge Components dispatch their collected data to the NOSTRADAMUS Cloud Platform through the IoT Platform, leveraging the Cloud-side API provided by the Edge Gateway.

Furthermore, the Low-Code Platform of NOSTRADAMUS empowers domain experts to describe their edge device characteristics and sensor configurations. The platform's application generation engine then automatically produces the necessary embedded software and connectivity code for these devices. This generated software enables the edge devices to dispatch data to the cloud via the Edge Gateway.

In summary, the Edge Gateway is a sophisticated, multi-protocol data ingress system that forms the vital link between the physical agricultural environment and the advanced cloud analytics of the NOSTRADAMUS platform. Its design reflects a deep understanding of the unique challenges of agricultural IoT data, prioritizing robust ingestion, low-latency processing, and seamless integration within a hybrid Edge-to-Cloud architecture.

Data Stream Layer: Apache Kafka

At the core of the architecture lies Apache Kafka, a distributed event streaming platform. Apache Kafka serves as the highly scalable, open-source distributed event streaming platform at the core of the IoT Observatory's Speed Layer. It is specifically engineered to manage the high velocity of agricultural data, capable of handling massive volumes of events with high throughput and low latency. Within the Nostradamus *System of Systems*, Kafka acts as the central message broker, facilitating the seamless integration of heterogeneous data systems and ensuring a reliable flow of data between ingestion points, processing engines, and persistent storage.

The Kafka infrastructure achieves this high-performance streaming through four primary functional components:

- **Topics:** Categories or feed names to which messages are sent and from which they are read.
- **Producers:** Applications (such as the Edge Gateway or REST API) that publish data to Kafka topics.
- **Consumers:** Applications that subscribe to specific topics and process the published data feeds.
- **Brokers:** The physical servers that store the data streams and serve the connected clients.

To ensure precise data management and multi-tenancy segmentation, Kafka topics in the Nostradamus platform adhere to a strict hierarchical naming convention: {organization_name}.{project_name}.{collection_name}. This allows the system to clearly isolate data originating from different demonstration sites or third-party applications.

Data Processing Layer: Flink and ksql-DB

To transform raw streams into insight, the architecture deploys two complementary processing engines:

- **ksqlDB (Stream Analytics):** This component democratizes stream processing by allowing developers to run SQL-like queries directly on infinite streams of data. It is utilized for filtering, transformation, and standard aggregations, simplifying real-time data manipulation without complex coding
- **Apache Flink (Complex Event Processing):** For advanced analytics, the system utilizes Flink. Flink provides unified stream and batch processing capabilities with advanced state management. Uniquely, NOSTRADAMUS leverages Flink to integrate Machine Learning (ML) directly into the stream, enabling real-time inference (e.g., pest detection models) and supporting the execution of custom user-defined scripts for adaptive learning

Data Persistence Layer: Apache Cassandra

For long-term storage, Apache Cassandra was selected for its decentralized, peer-to-peer architecture. Unlike master-slave databases, Cassandra has no single point of failure, offering 99.99% availability. Its ability to scale linearly, increasing capacity simply by adding nodes, makes it the ideal repository for the high volume of time-series data generated by the pilot sites

Key aspects of Cassandra's implementation in NOSTRADAMUS include:

- **High Availability and Fault Tolerance**: Data redundancy is ensured through replication across multiple nodes.
- **Flexible Data Model**: Cassandra provides flexibility for various data types and supports dynamic and scalable schemas.
- **Data Organization**: Data is structured hierarchically.
    - Keyspaces: Each organization (e.g., a demonstration site or third party) is isolated within its own Cassandra Keyspace, ensuring data segregation and named after the organization.

    - Column Families (Tables): Within each keyspace, collection data is stored in tables named {project_name}_{collection_name}.
    - Events: The fundamental unit of agricultural data, consisting of a timestamp and various attributes (e.g., soil pH, moisture).
- **Time-Series Storage Strategy (Bucketing)**: To optimize read and write performance for high-volume, high-velocity IoT data, a partitioning and bucketing strategy is used.
    - Partition Key: Each record is partitioned using a unique source identifier (e.g., sensor ID) and a time bucket (typically daily). This co-locates all readings from the same sensor for a specific day, drastically reducing query latency for time-series analysis.
    - Clustering Key: The event timestamp serves as the clustering key, defining the sort order within each partition.
- **Metadata Management**: A specialized metadata keyspace in Cassandra manages relational information about users, organizations, projects, and collections, enhancing data retrieval and accessibility.

4.2.3 SERVING LAYER

The Serving Layer acts as the unified interface for end-users and applications, merging the real-time views from the Speed Layer with the batch views from the Batch Layer. This provides a seamless query experience, abstracting whether data originates from memory (speed) or disk (batch).

The Serving Layer's functions include:

- **Interface for Insights**: It provides actionable insights and functionalities to end-users through an API that supports specific data queries, as well as enabling dashboards and mobile applications.
- **Low-Latency Access**: By leveraging processed data stored in Apache Cassandra, it ensures fast and efficient data retrieval for applications requiring low-latency access.
- **Ad-Hoc Querying**: The system allows for user-defined ad-hoc queries, which can be calculated in real-time, providing flexibility in application development. The stream processing layer is instrumental in calculating these queries.
- **Data Flow Integration**: Data flows from ingestion through Kafka to ksqlDB and Apache Flink for real-time processing. Results are fed back into Kafka and consumed by the Serving Layer. Simultaneously, Kafka routes raw data and processing results to Cassandra for long-term storage, which is then accessed by the Serving Layer based on user needs.

4.2.4 DATA SCHEMA AND HIERARCHY

The IoT Observatory (II) implements a structured, hierarchical abstraction layer to organize and manage event-based data effectively. This hierarchy is designed to ensure multi-tenancy, data segregation, and high-performance retrieval across the various European pilot sites and third-party applications. The data schema is organized into four primary levels: a) Organizations, b) Projects, c) Collections, and d) Events.

A. Organizational Hierarchy

- **Organizations:** This represents the top-level entity within the system. In the context of NOSTRADAMUS, separate organizations are created for each demonstration site (e.g., Cyprus, Slovenia), core module development partners, and funded third parties. Organizations act as the primary security boundary, managing and controlling access to their internal resources to ensure strict privacy.
- **Projects:** Within an organization, data is further partitioned into Projects. A project has a defined scope and contains the specific resources (collections) needed to achieve its objectives. For example, a pilot site might maintain separate projects for different geographic regions or experimental domains.
- **Collections:** A collection is a structured set of data within a project, functioning similarly to a table in a relational database. Each collection follows a defined schema, allowing for granular tracking of specific research contexts, such as an "IoT Sensor Dataset" or "Meteorological Records."
- **Events:** The event is the fundamental unit of agricultural data. Each event consists of a timestamp (the moment of creation) and various attributes providing domain-specific information, such as soil pH, moisture, or temperature readings. The system supports automated schema identification to ensure incoming events conform to the expected formats.

This hierarchical structure in the name allows clear and precise data management and segmentation. In Cassandra, every organization has its own keyspace (named by the organization), ensuring data segregation for the same organization's data. Within each keyspace, the collections data are stored in column families following the naming convention {project_name}_{collection_name}. This convention provides clear and organized data storage, reflecting the hierarchical structure of projects and collections within the organization. The columns of each column family refer to the attributes of the events. The partition key of each record consists of the key and a time bucket, such as the day. In Cassandra, bucketing data involves organizing data into manageable partitions to optimize read and write performance. This technique is particularly useful for time-series data, where large volumes of data are continuously ingested. By using daily time buckets, the records referring to the same key and day will be stored in the same partition, improving read performance. The partition key, along with the timestamp, which is the clustering key, form the primary key (e.g. sensor1, 2024-06-25, 2024-06-25T00:00:00Z, 2024-06-24T00:10:01Z). Figure 6 depicts the schema of the infrastructure.

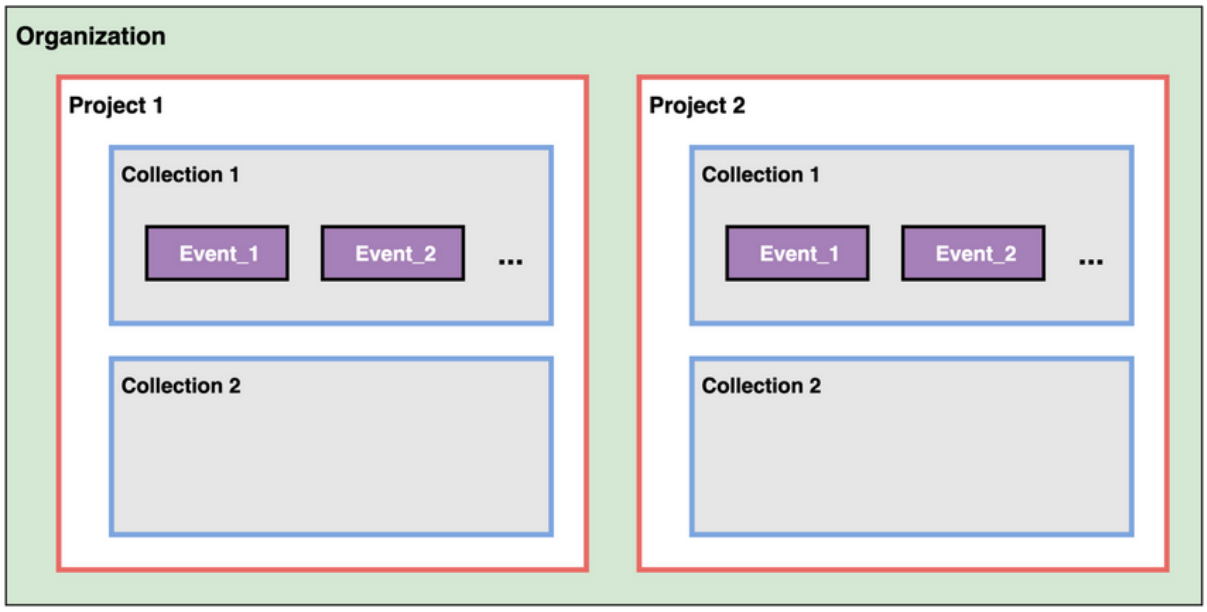

FIGURE 6: IOT OBSERVATORY METADATA SCHEMA

Furthermore, Figure 7 depicts an example for a project that is stored using the data schema.

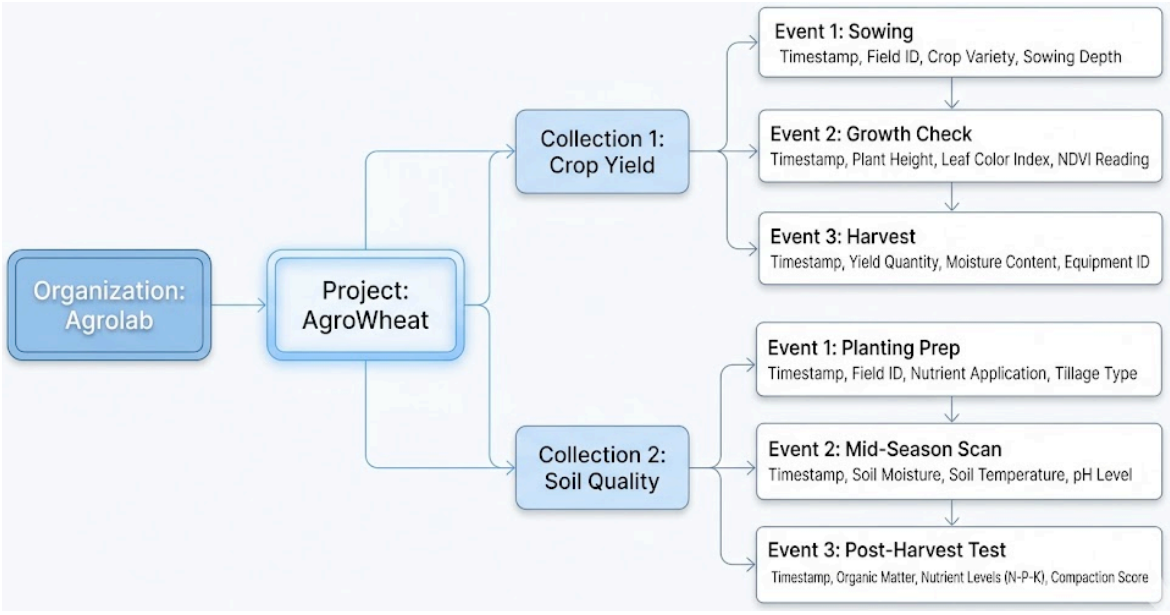

FIGURE 7: EXAMPLE OF A MULTI-COLLECTION PROJECT IN IOTO

Additionally, a metadata keyspace is created in Cassandra, which serves as a foundational element for efficiently managing and organizing the system's metadata. This keyspace is designed to capture essential information about users, organizations, projects, and collections, making it easier to find and understand the relationships between different data entities. By structuring the data into distinct tables for users, organizations, projects, and collections, the system can store detailed

metadata, including creation dates and tags, which enhances the ability to search, filter, and analyze the data.

B. Technical Mapping and Naming Conventions

To maintain consistency across the distributed system, the hierarchy is directly mapped to the underlying message buffering and storage layers using strict naming conventions:

- **Apache Kafka (Data Streams):** Data for each collection is routed to a dedicated topic following the format:
  {organization_name}.{project_name}.{collection_name}
- **Apache Cassandra (Storage):**
  - Keyspaces: Each organization is isolated within its own Cassandra Keyspace, named after the organization.
  - Column Families (Tables): Within a keyspace, collection data is stored in tables named:
    {project_name}_{collection_name}.

C. Time-Series Storage Strategy

Given the "High Volume" and "High Velocity" nature of agricultural IoT data, the system utilizes a partitioning and bucketing strategy in Cassandra to optimize read and write performance.

- **Partition Key:** Each record is partitioned using a combination of a unique source identifier (e.g., a specific sensor ID) and a time bucket (typically daily). This ensures that all readings from the same sensor for a specific day are stored physically together on disk, drastically reducing query latency for time-series analysis.
- **Clustering Key:** The event timestamp serves as the clustering key, defining the sort order within each partition.

D. Metadata Management

Beyond the raw event data, a specialized metadata keyspace is maintained in Cassandra. This keyspace manages relational information about users, organization details, creation dates, and metadata tags. By linking projects to their parent organizations and collections to their specific projects, the system provides a comprehensive, searchable view of the data landscape, facilitating advanced decision-making and cross-entity analysis.

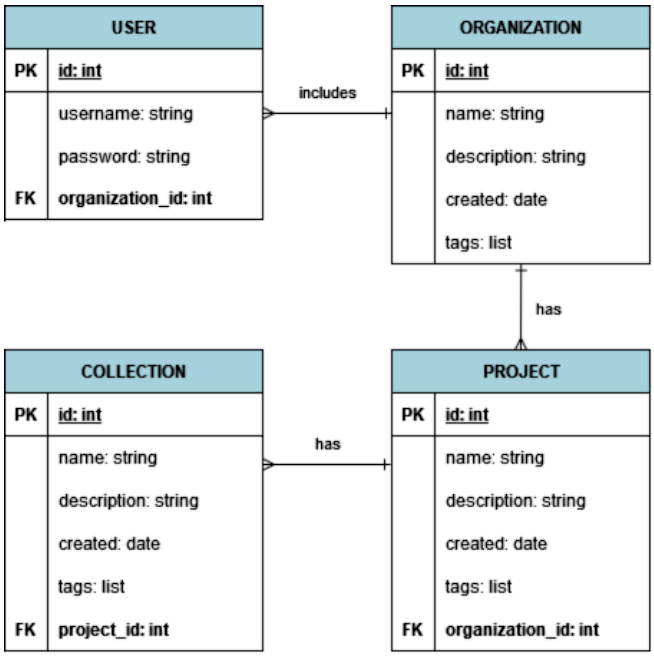

FIGURE 8: IOTO METADATA MODEL

This metadata organization approach not only improves data retrieval and accessibility but also helps uncover valuable connections and insights. For instance, by linking projects to their respective organizations and collections to their projects, the system can provide a comprehensive view of how different components are related. This facilitates better decision-making, more efficient data management, and a deeper understanding of the data landscape within the organization.

4.2.5 DATA FLOW ACTIVITIES

This section indicates the data flows in the IoTO, providing a set of possible paths to better identify the different data movements and interactions within the system.

Data Flow 1 – Data Ingestion

- **Description:** Input data is sent to the system via the API. This data is then published to a Kafka topic and is stored in a column family in Cassandra.
- **Process:**
  - **Data Ingestion:** Users, IoT and Edge devices, or external systems send raw data to the system through a RESTful API endpoint.
  - **Send to Kafka:** The API acts as a Kafka producer and publishes the incoming data to a specific Kafka topic.
  - **Store in Cassandra:** A Kafka consumer listens to this topic and writes the data to the corresponding column family in Cassandra.

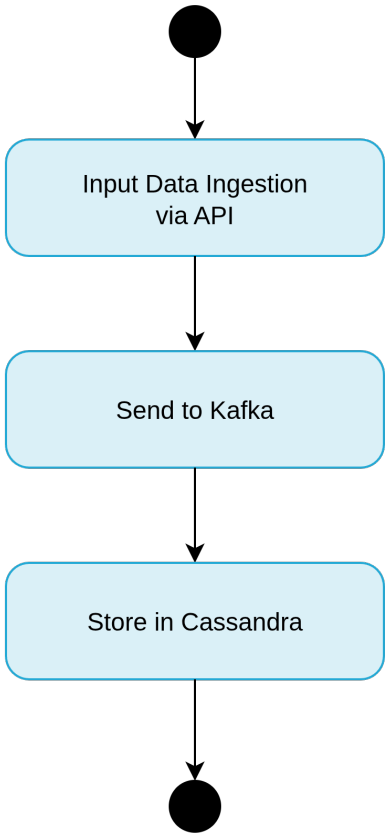

FIGURE 9: DATA FLOW 1 – DATA INGESTION

Data Flow 2 - Historical Data Retrieval

- **Description:** Users can request specific historical data from a collection, using filters to narrow down the results. The system retrieves this data from Cassandra and applies the necessary filters.
- **Process:**
  - **User Request:** The user sends a query request to the system, specifying filters and criteria.
  - **Query Processing:** The system translates these filters into a Cassandra query.
  - **Data Retrieval:** The system executes the generated Cassandra query and retrieves the filtered historical data.
  - **Further Data Filtering:** The retrieved data is further filtered (if needed) in the serving layer to match the user's request.
  - **Results Delivery:** The processed data is returned to the user, providing the requested historical data.

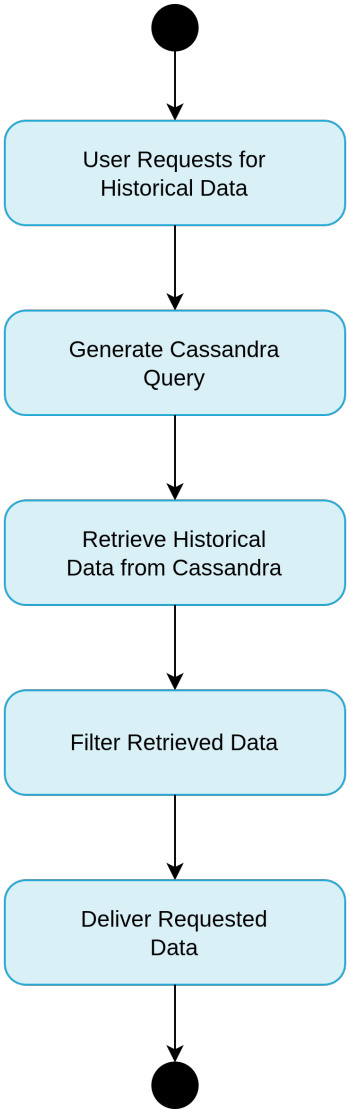

FIGURE 10: DATA FLOW 2 - HISTORICAL DATA RETRIEVAL

Data Flow 3 - Read Live Data Stream

- **Description:** Users can request a live view of a collection's data. A Kafka consumer instance subscribes to the relevant Kafka topic and streams new events to the user.
- **Process:**
  - **User's Live Data Request:** The user initiates a subscription request for a live data feed of a specific collection.
  - **Kafka Consumer Initialization:** A Kafka consumer instance is created and subscribes to the specified Kafka topic.
  - **Receiving Events:** The Kafka consumer continuously listens to the subscribed Kafka topic for new events. As new events are published to the Kafka topic, the consumer receives these events in real-time.

- **Streaming Events to User:** The Kafka consumer streams the received events directly to the user, providing a real-time live view of the collection data.

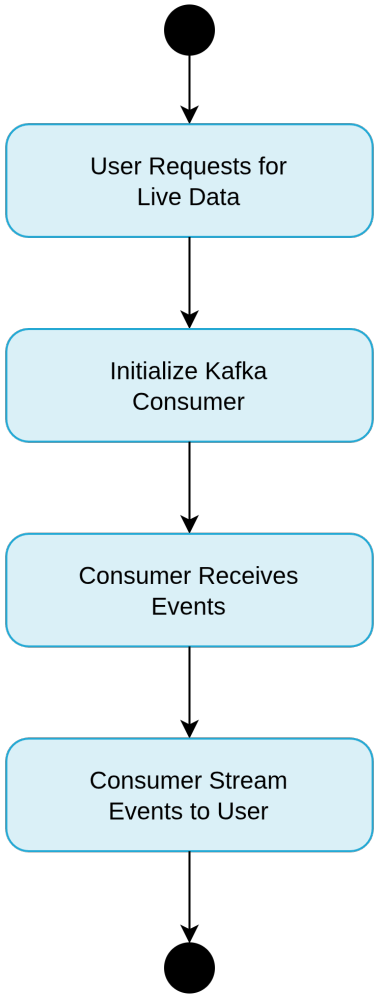

FIGURE 11: DATA FLOW 3 - READ LIVE DATA STREAM

Data Flow 4 - Historical Data Aggregation

- **Description:** Users can request an aggregation of historical data from a collection. The system retrieves relevant data from Cassandra and performs the aggregation.
- **Process**:
  - **User Requests Aggregation:** The user sends a request specifying the type of aggregation needed (e.g., sum, average, count) along with any necessary filters and criteria.
  - **Query Processing:** The system translates the user's request into a Cassandra query.
  - **Data Retrieval:** The system executes the query to retrieve the relevant historical data from Cassandra.
  - **Result Delivery:** The aggregated result is returned to the user.

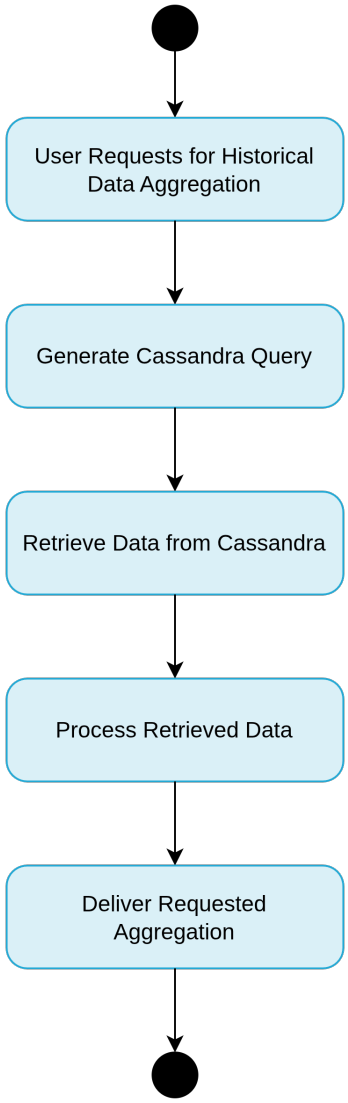

FIGURE 12: DATA FLOW 4 - HISTORICAL DATA AGGREGATION

Data Flow 5 - Real-Time Data Aggregation

- **Description**: Users can request real-time data aggregation. Depending on the complexity, the system uses ksqlDB for simple aggregations or Flink for more complex ones. The results are sent to a Kafka topic and stored in Cassandra.
- **Process**:
  - **User Requests Real-Time Aggregation**: The user specifies the real-time aggregation needed, including the necessary filters and criteria.
  - **Aggregation Specification**: The system determines whether the aggregation is simple or complex, using ksqlDB for simpler aggregations like averaging a single attribute and Flink for more complex aggregations (e.g. involving machine learning).

- **Get Data from Kafka:** The selected tool subscribes to the relevant Kafka topic to get the necessary information for the aggregation of real-time data.
- **Aggregation**: The selected tool performs the real-time aggregation.
- **Result Streaming**: The aggregation results are published on a Kafka topic.
- **Data Storage**: The results are also stored by a Kafka consumer in Cassandra for further use.

To provide further clarity for all readers, particularly those with a less technical background, the distinction between "simple" and "complex" aggregation within Nostradamus is critical.

Simple Aggregations (using ksqlDB) typically involve straightforward, SQL-like queries applied to a single stream of data, focusing on fundamental statistical summaries. Examples include calculating the average, sum, count, minimum, or maximum of a specific sensor reading over a defined period (e.g., the average temperature from a particular sensor over the last hour, or total daily rainfall). ksqlDB is ideal for democratizing stream processing by allowing users to perform these queries directly on continuous data streams without requiring extensive coding expertise.

Complex Aggregations (using Apache Flink) go beyond basic statistical summaries on single data attributes, leveraging advanced processing capabilities. Flink is employed for these advanced analytics, which often integrate multiple data sources, apply sophisticated algorithms, or handle intricate temporal and spatial relationships:

- **Multi-dimensional Aggregations:** This involves combining data across various dimensions such as time, space, and other criteria.
  - **Time-based:** Aggregating data over flexible and extended time windows (e.g., 1, 5, or 10 years, or custom intervals such as "every 5 days"). Flink's capabilities in event-time processing and windowing are crucial for accurate time-series analysis. The underlying Cassandra database, with its time-series storage strategy and data bucketing (e.g., grouping data by sensor ID and daily time buckets), efficiently supports such complex temporal queries.
  - **Space-based:** Aggregating data relevant to specific geographical areas (e.g., a single pixel, a particular field region, a country, a larger union, or even a continent). This often requires integrating high-velocity IoT streams from the IoT Observatory Layer with harmonized geospatial data from the Data Infrastructure Layer's Data Cubes. These Data Cubes provide multi-dimensional arrays for organizing geospatial data, making sophisticated spatial aggregation feasible.

- **Advanced Statistical and Analytical Operations:** Moving beyond simple sums or averages to incorporate more sophisticated statistical computations like median, standard deviation, variance, or other custom analytical calculations.

- **Machine Learning Integration:** Flink can directly integrate machine learning (ML) models into the data stream, enabling real-time inference (e.g., immediate pest detection based on sensor data patterns) and supporting adaptive learning. This is vital for transforming raw data into high-level indicators and complex predictions within domain-specific algorithms.

- **Custom Logic Execution:** The system explicitly allows users to upload custom scripts (e.g., Python) to perform specific, user-defined aggregations on live data streams via the processing engine. This capability is instrumental in transforming raw data into actionable intelligence by applying unique, domain-specific algorithms.

In essence, while simple aggregations provide quick statistical snapshots, complex aggregations within Nostradamus, predominantly facilitated by Apache Flink, deliver deeper, multi-faceted insights essential for informed decision-making in European agriculture. This functionality, including advanced socio-economic analysis and spatio-temporal data manipulation, will be made accessible to experts through the platform's Low-Code Application Layer, which abstracts the underlying technical complexities via Domain-Specific Languages.

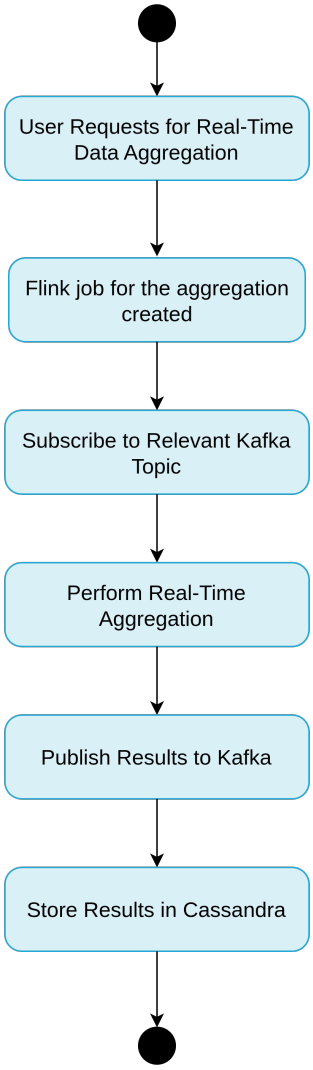

FIGURE 13: DATA FLOW 5 - REAL-TIME DATA AGGREGATION

Data Flow 6 – Register Historical Data Aggregation

- **Description**: Users can send a custom aggregation file to the system, which retrieves relevant historical data from Cassandra and performs the aggregation.
- **Process:**
  - **User File Submission**: The user uploads a file containing the custom aggregation logic.
  - **Data Retrieval:** The system queries Cassandra to retrieve the relevant historical data based on the criteria specified in the custom aggregation file.
  - **Aggregation:** The system performs the custom aggregation using the logic provided in the user-uploaded file.
  - **Results Delivery:** The aggregation results are returned to the user.

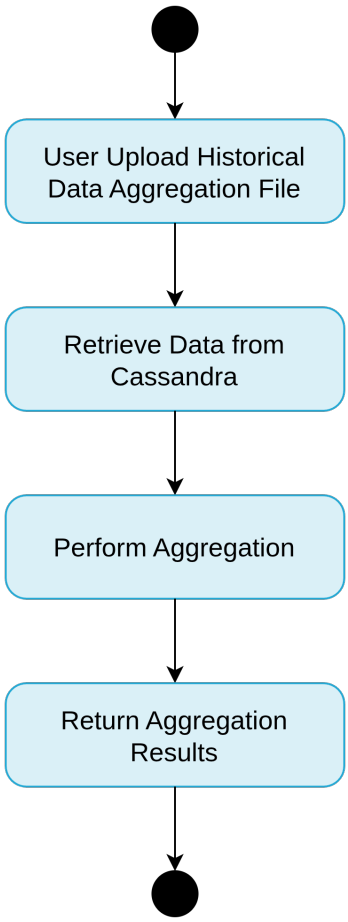

FIGURE 14: DATA FLOW 6 – REGISTER HISTORICAL DATA AGGREGATION

Data Flow 7 – Read Historical Data Aggregation

- **Description**: Users can send a custom aggregation file to the system, which uses Flink to perform the real-time aggregation. The results are sent to Kafka and stored in Cassandra.
- **Process:**
  - **User File Submission**: The user uploads a file containing the custom aggregation logic.
  - **Get Data from Kafka:** The selected tool subscribes to the relevant Kafka topic to get the necessary information for the aggregation of live data.
  - **Aggregation**: The system uses Flink to implement and execute custom aggregation based on the user-provided file.
  - **Result Streaming**: The results of the custom aggregation are sent to a Kafka topic.
  - **Data Storage**: The results are stored in Cassandra for further use.

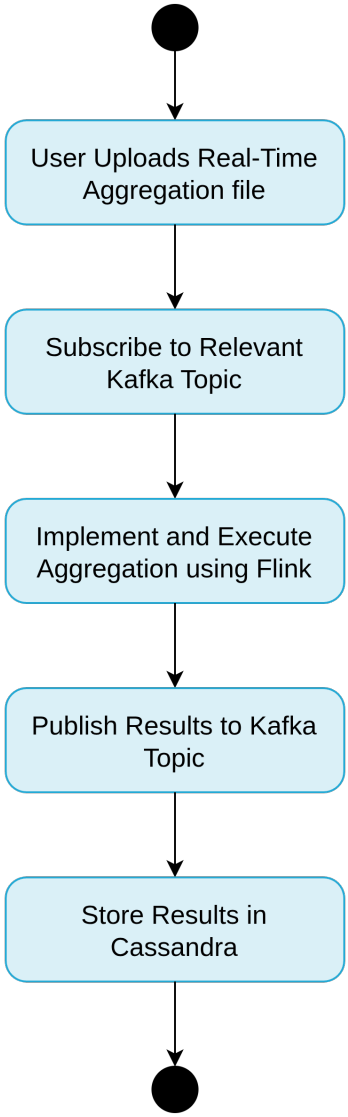

FIGURE 15: DATA FLOW 7 – READ HISTORICAL DATA AGGREGATION

4.2.6 SECURITY & ACCESS-CONTROL CONSIDERATIONS

The security and access control mechanisms within the NOSTRADAMUS IoT Observatory are designed to ensure the confidentiality, integrity, and controlled availability of agricultural data, which is crucial for the project's success in promoting food security and agricultural resilience. This section outlines the comprehensive framework that protects the system from unauthorized access and ensures robust data governance across the hybrid Edge-to-Cloud architecture.

The IoT Observatory implements a rigorous **Attribute-Based Access Control (ABAC)** mechanism to manage user roles and data access. This approach allows for granular control over who can interact with specific data and functionalities. There are three primary roles defined within the system:

- **Admin:** Administrators possess full access to all data and services, including the capability to perform aggregations. They are also responsible for the critical task of creating and validating new users, thereby ensuring that only authorized personnel can access sensitive information.
- **User:** Typically representing researchers from an organization (e.g., a Living Lab), users have read/write access limited to their specific projects within the observatory. This role fosters collaboration and contribution to ongoing research efforts, allowing full utilization of tools and data within their defined project scope.
- **Guest:** This role is designated for external stakeholders and provides read-only access to public data. Guest users can view and analyze data relevant to their needs without the ability to modify it, making it suitable for requesting public data for dashboards or interactive maps.

To ensure a secure and reliable platform, the observatory employs secure tunnelling for user creation and authentication. The infrastructure uses HTTPS to encrypt data exchanged between a user and the system. This encryption safeguards sensitive user data from interception and unauthorized access, ensuring that all interactions are secure.

User access is optimized by using access tokens instead of traditional basic authentication mechanisms. Authentication and authorization of requests on the platform are achieved using API tokens and JWT tokens. This approach enhances security by minimizing the exposure of sensitive credentials, thereby reducing the risk of unauthorized access. These measures collectively ensure that data transmission is secure, and authentication processes are robust, protecting both the users and the integrity of the observatory data.

For the data themselves, API keys are utilized to authorize requests. There are three main keys that manage access to a project, generated from the corresponding organization:

1. **Master Key:** The most powerful key for each project, allowing authentication and authorization for any request, including administrative tasks such as data deletion.
2. **Write Key:** A key focused on enabling the recording of events in a project.
3. **Read Key:** A key dedicated to retrieving information from a project.

Security will be in place to validate various authentication and authorization scenarios, ensuring robust security measures across the system.

Furthermore, data segregation is a critical non-functional requirement to maintain multi-tenancy privacy within the system. The IoT Observatory enforces strict isolation between different organizations and projects through a structured, hierarchical abstraction layer.

Beyond internal access control, the system also considers Secure Asset Access and Secure Data Transport:

- **Secure Asset Access:** Leverages ABAC methods to strictly control access, ensuring that only authenticated and authorized entities can interact with the data.
- **Secure Data Transport:** As noted, all data exchanged between users and the system is encrypted using HTTPS/TLS, and the Edge Gateway also facilitates secure communication from IoT devices.

The security architecture of the NOSTRADAMUS IoT Observatory is visually represented in Figure 16, which illustrates our multi-layered approach to data protection across the hybrid Edge-to-Cloud continuum. The framework is initiated by authenticating users, categorized as Admins, Users, or Edge Devices, through a secure HTTPS/TLS tunnel using modern JWT and API tokens to prevent credential exposure. All access requests are then processed by a central Attribute-Based Access Control (ABAC) engine, which enforces granular permissions based on the user's role and project affiliation. This ensures that access is strictly controlled, aligning with the principle of least privilege. Furthermore, the diagram depicts the system's robust multi-tenancy, showcasing how data is segregated into distinct organizations and projects, with programmatic access governed by a tiered system of Master, Write, and Read API keys. This comprehensive design guarantees the confidentiality, integrity, and controlled availability of sensitive agricultural data, securing the platform from unauthorized access while enabling legitimate research and analysis.

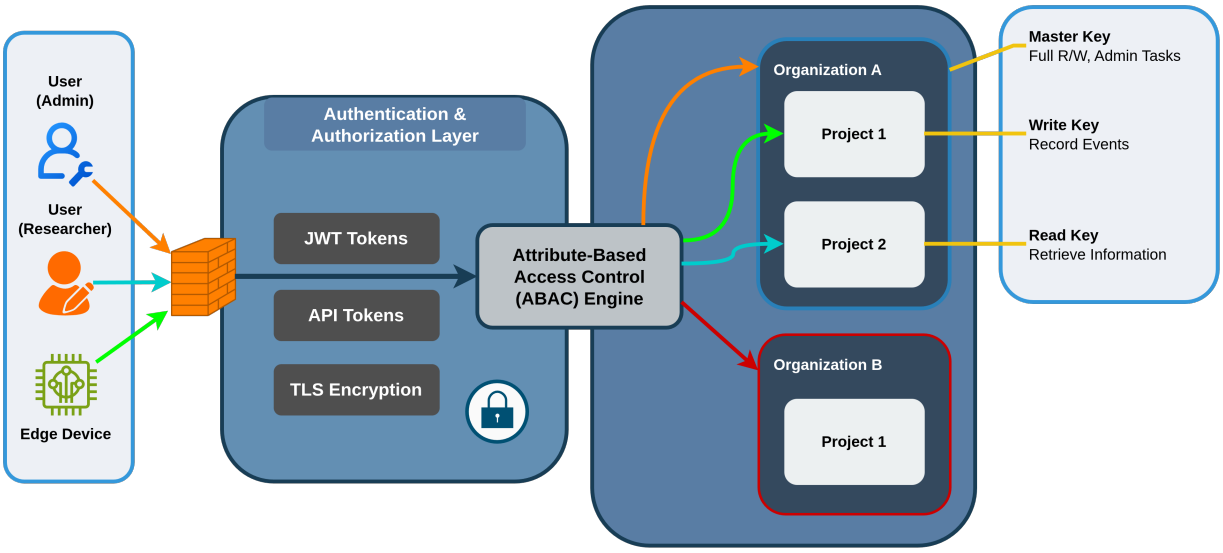

FIGURE 16: IOT OBSERVATORY SECURITY ARCHITECTURE

By integrating these robust security and access control features, the IoT Observatory ensures a secure and reliable environment for managing the critical agricultural data streams essential to the NOSTRADAMUS project, supporting both real-time decision-making and long-term analysis across the entire Edge-to-Cloud continuum.

4.2.7 INTERFACE SPECIFICATIONS

To ensure broad interoperability and ease of integration for third-party developers (WP6), the IoT Observatory exposes a comprehensive REST API that is compliant with OpenAPI v3 specifications. The specification file can be distributed to partners and third parties to enable the rapid development of software that interacts with real-time and historical IoT data, thanks to the tools provided for automatically generating client and server library stubs for various programming languages.

There are two types of API endpoint: those that require a user token to perform management actions (e.g. creating an organization or project) and those that use project API keys to perform data operations (e.g. sending events, retrieving historical or real-time data, performing aggregations, etc.).

The OpenAPI specifications (YAML file) of the NOSTRADAMUS IoT Observatory REST API are available via the following link:

NOSTRADAMUS_ioto_restapi_specs_v1[1]

TABLE 1 REST API ENDPOINTS

API Call Id/Name	API Call
1. Create Organization	**POST** /api/organizations
2. Update Organization Info	**PUT** /api/organizations/{ORG_NAME}
3. Delete Organization	**DELETE** /api/organizations/{ORG_NAME}
4. Get Organization Information	**GET** /api/organizations/{ORG_NAME}
5. Get All Organizations	**GET** /api/organizations
6. Create User	**POST** /api/organizations/{ORG_NAME}/users
7. Delete User	**DELETE** /api/organizations/{ORG_NAME}/users/{USERNAME}
8. Update User Password	**PUT** /api/organizations/{ORG_NAME}/users/{USERNAME}
9. Get All Users of Organization	**GET** /api/organizations/{ORG_NAME}/users
10. Create project	**POST** /api/organizations/{ORG_NAME}/projects
11. Update Project	**PUT** /api/organizations/{ORG_NAME}/projects/{PROJECT_NAME}
11. Delete Project	**DELETE** /api/organizations/{ORG_NAME}/projects/{PROJECT_NAME}

12. Get All Projects of an Organization	**GET** /api/organizations/{ORG_NAME}/projects
13. Get Information of a Project	**GET** /api/organizations/{ORG_NAME}/projects/{PROJECT_NAME}
14. Regenerate Key of a Project	**POST** /api/organizations/{ORG_NAME}/projects/{PROJECT_NAME}/keys
15. Get Key of a Project	**GET** /api/organizations/{ORG_NAME}/projects/{PROJECT_NAME}/keys
16. Create Collection	**POST** /api/organizations/{ORG_NAME}/projects/{PROJECT_NAME}/collections
17. Update Collection Info	**PUT** /api/organizations/{ORG_NAME}/projects/{PROJECT_NAME}/collections/{COLLECTION_NAME}
18. Get all Project Collections	**GET** /api/organizations/{ORG_NAME}/projects/{PROJECT_NAME}/collections
19. Get Collection Information	**GET** /api/organizations/{ORG_NAME}/projects/{PROJECT_NAME}/collections/{COLLECTION_NAME}
20. Delete Collection	**DELETE** /api/organizations/{ORG_NAME}/projects/{PROJECT_NAME}/collections/{COLLECTION_NAME}
21. Send Data to Collection	**POST** /api/organizations/{ORG_NAME}/projects/{PROJECT_NAME}/collections/{COLLECTION_NAME}/send_data
22. Get Collection Data	**GET** /api/organizations/{ORG_NAME}/projects/{PROJECT_NAME}/collections/{COLLECTION_NAME}/get_data
23. Get Collection Live Data	**POST** /api/organizations/{ORG_NAME}/projects/{PROJECT_NAME}/collections/{COLLECTION_NAME}/live_data
24. Get Collection Statistics	**GET** /api/organizations/{ORG_NAME}/projects/{PROJECT_NAME}/collections/{COLLECTION_NAME}/statistics
25. Post Live Aggregation	**POST** /api/organizations/{ORG_NAME}/projects/{PROJECT_NAME}/collections/{COLLECTION_NAME}/live-aggregation
26. Get Custom Aggregation	**GET** /api/organizations/{ORG_NAME}/projects/{PROJECT_NAME}/aggregations
27. POST Custom Live Aggregation	**POST** /api/organizations/{ORG_NAME}/projects/{PROJECT_NAME}/live-aggregations

4.2.8 SOFTWARE DEVELOPMENT KIT

The NOSTRADAMUS IoTO Python SDK[5] is a client library designed to facilitate interaction with the NOSTRADAMUS IoT Observatory API. Its primary purpose is to simplify the management of IoT data collection, projects, and analytics within the broader NOSTRADAMUS platform. This SDK acts as a crucial interface, allowing developers and domain experts to programmatically control and access the robust data management capabilities of the IoT Observatory.

The SDK directly interacts with the IoT Observatory, which is built upon technologies like Apache Kafka for data streams and Cassandra for storage. The SDK's functionalities, such as sending data and querying, directly correspond to the data flow activities and serving layer capabilities of the IoT Observatory. This SDK simplifies access to the IoT Observatory's core features, which are essential for processing high-velocity sensor data streams and managing a hierarchical data structure of organizations, projects, and collections, thus enabling developers to build digital solutions for agricultural resilience and sustainability more effectively.

To support the rapid development of applications interacting with IoTO, the SDK offers a comprehensive set of features engineered for resilience and ease of use in production environments:

- **Synchronous and Asynchronous Support:** It supports both synchronous and asynchronous operations, allowing developers to choose the client type that best suits their application needs.
- **Type Hints & Validation:** Includes full type coverage utilizing Pydantic models, which enhances code reliability and developer experience.
- **Automatic Retry & Rate Limiting:** Built-in mechanisms for automatic retries and rate limiting ensure resilience against temporary network issues or API call limits, contributing to stable operation.
- **Dual Authentication:** It supports two primary authentication methods: OAuth2 and API Key authentication, aligning with the security considerations of the IoT Observatory.
- **Comprehensive Error Handling:** Provides clear and actionable error messages, aiding in debugging and robust application development.
- **CLI Tool Included:** A command-line interface (CLI) tool is part of the SDK, enabling common operations to be performed directly from the terminal.
- **Documentation and Testing:** Well-documented with examples and has undergone thorough testing, boasting over 94% code coverage, ensuring reliability and maintainability.

Furthermore, the SDK provides comprehensive coverage for the NOSTRADAMUS IoTO API endpoints, allowing for a wide range of interactions:

- **Organizations:** Provides operations to get and update organization information.
- **Projects:** Offers full Create, Read, Update, and Delete (CRUD) operations for projects.

[5] https://github.com/robotics-4-all/NOSTRADAMUS-ioto-sdk

- **Project Keys:** Enables management of API keys, including read, write, and master keys.
- **Collections:** Supports full CRUD operations for data collections within projects.
- **Data Operations:** Facilitates sending, querying, and deleting data, with options for filtering and aggregations.
- **Statistics:** Allows retrieval of analytics with simple and complex aggregations (previously defined in sections 4.2.2 and 4.2.5).

Finally, IoTO Python SDK is an open-source project, currently hosted on github[6], and distributed under the MIT License.

4.3 DATA CUBES

The Data Cubes (IV) represent the centralized data-based silo and the analytical backbone of the NOSTRADAMUS Cloud Platform. They are engineered as multi-dimensional arrays that organize Earth Observation (EO) archives, rasterized IoT telemetry, and socio-economic indicators across dimensions of space, time, and data type. By providing a structured and efficient way to store and process large amounts of previously scattered data, the Data Cubes serve as an open-source "one-stop-shop" that promotes the responsible use of digital technologies in European agriculture.

The implementation follows the Open Data Cube (ODC) framework, which is specifically designed to unleash the power of Big Earth Data by providing Analysis Ready Data (ARD). To ensure the system is both scalable and tailored to the diverse biogeographical conditions of the EU, the project utilizes a Data Cube on Demand (DCoD) approach. This allows for the rapid instantiation of five dedicated Data Cubes for the project's demonstration sites in Cyprus, Germany, Switzerland, Slovenia, and Serbia, enabling contextualized discovery and high-performance "pixel-drilling" for spatiotemporal analysis.

Aligned with the FAIR data principles, these cubes decouple complex metadata from actual binary assets, providing standard interfaces for both human exploration via the Data Cube GUI and machine-to-machine interaction through the Datacube API. Ultimately, the Data Cubes act as the essential provisioning environment for the project's four core agricultural modules, transforming raw measurements into the high-veracity indicators required for drought monitoring, crop suitability estimation, soil fertility management, and sustainable pest control.

4.3.1 SYSTEM ARCHITECTURE

Figure 17 provides a comprehensive architectural overview of the NOSTRADAMUS Data Cube ecosystem, illustrating the end-to-end process from data ingestion to application-driven analysis. On the left, raw data inputs, including EO data, rasterized IoT telemetry, and socio-economic indicators, serve as the foundational assets. These inputs are processed through a central architecture built on the Open Data Cube framework, which logically separates the physical storage

[6] https://github.com/robotics-4-all/NOSTRADAMUS-ioto-sdk

of binary assets from the PostgreSQL-based metadata index, enabling efficient querying of ARD. The entire environment is encapsulated using a **Cube-in-a-Box (CiaB)** approach with Docker, which facilitates the rapid, on-demand deployment of dedicated Data Cubes for the project's pilot sites. On the right, the diagram shows the dual access interfaces: a machine-to-machine Datacube API that includes a STAC-compliant endpoint for interoperability, and a human-centric ODC Explorer Web UI and JupyterHub workspace for interactive exploration. Ultimately, this integrated system transforms raw measurements into high-veracity indicators that provision the four core agricultural modules.

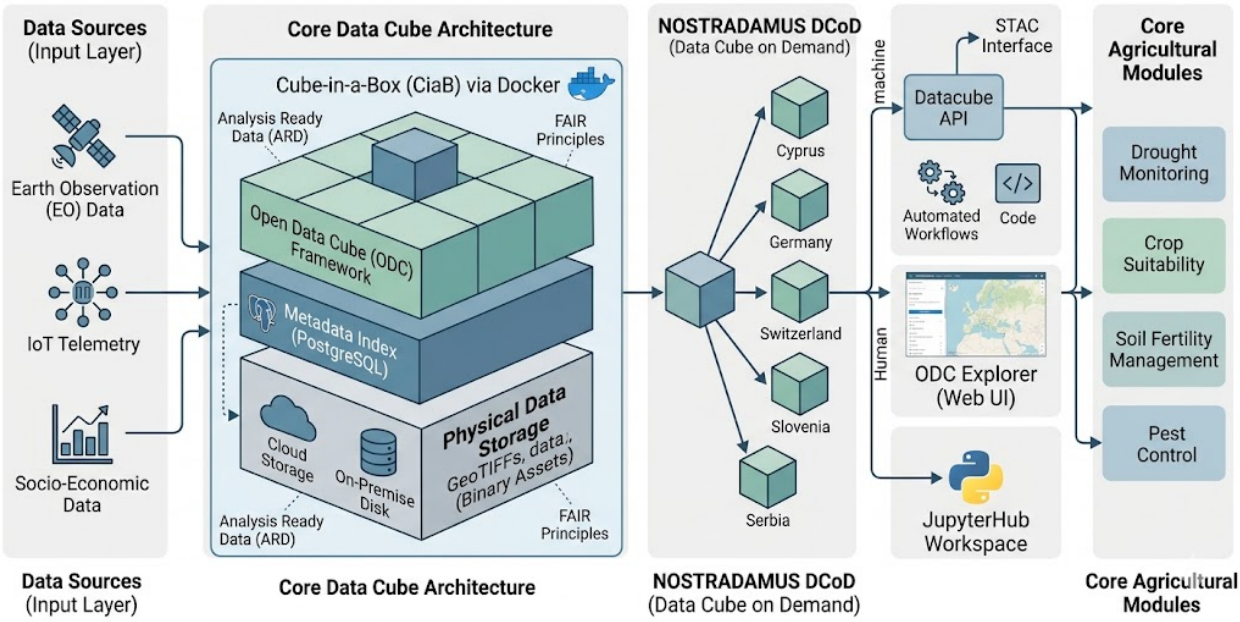

FIGURE 17: NOSTRADAMUS DATA CUBE ARCHITECTURE

Furthermore, it is important to highlight that the Datacube API is the primary interface through which the agricultural applications, generated from the Application Generation Engine, will programmatically access and process spatiotemporal data. This direct API integration ensures a seamless and automated workflow, allowing the custom-built applications to leverage the full analytical power of the Data Cube ecosystem for tasks such as drought monitoring and crop suitability analysis.

4.3.2 EO DATA CUBES

The rapid growth in the availability of EO data from satellite missions has created unprecedented opportunities for monitoring and understanding environmental and socio-economic processes. Programs such as the European Space Agency's Copernicus initiative and NASA's Earth observation missions continuously generate vast amounts of high-resolution and multi-temporal data. However, the massive volume and complexity of these datasets present significant challenges in terms of storage, processing, and accessibility, particularly for users with limited technical infrastructure.

As the need for novel technologies and approaches to deal with Big Earth Data management and analysis has been recognized, several solutions and platforms have been proposed and developed based on the 'moving code' paradigm. Organizations providing these solutions have developed cloud-computing platforms such as the Amazon Web Services, the European Commission-funded Data and Information Access Services (DIAS) and Destination Earth (DestinE), Google Earth Engine, the System for Earth Observation Data Access Processing and Analysis for Land Monitoring (SEPAL) and Microsoft's Planetary Computer. These platforms continue to gather interest as they allow simple access to Big Earth Observation Data and concurrently provide high-performance computing resources and tools to process these via the Internet. DestinE, in particular, represents a major step forward in addressing several limitations of traditional proprietary EO platforms. By integrating large-scale Earth system modelling, high-performance computing, and data access within a unified environment, it promotes improved interoperability, enhanced access to harmonized datasets, and more transparent and collaborative approaches to Earth system analysis. In this sense, DestinE moves towards more open and integrated digital infrastructures for Earth system monitoring and prediction. However, most currently operating cloud-computing platforms, including DestinE to some degree, remain centrally managed systems and may rely on controlled or semi-closed software ecosystems. This can limit the ability of external users to freely extend internal processing environments with new tools, workflows, or experimental methods. In addition, while DestinE promotes openness and interoperability, its large-scale, centrally coordinated architecture and evolving governance structure can constrain local adaptability and independent system evolution. Compared to more decentralized approaches, it offers less flexibility for fully autonomous deployment and user-driven modification outside the core infrastructure. Furthermore, as a continuously evolving European-scale initiative, its long-term functionality and feature set remain dependent on central development roadmaps and institutional priorities.

EO Data Cubes have emerged as a response to many of the structural and governance limitations observed in major cloud-based Earth observation platforms, including the constraints described above. Contrary to these platforms, EO Data Cubes are typically based on open standards and open-source frameworks (such as Open Data Cube), which directly addresses several of these challenges. One key advantage of EO Data Cubes is their openness and extensibility. Unlike centrally managed platforms, they allow users to integrate custom algorithms, models, and workflows directly into the system. This makes it easier to incorporate regional datasets, domain-specific tools, and locally developed methodologies, which is essential for tailoring analyses to specific environmental or socio-economic contexts. EO Data Cubes also improve interoperability by relying on standardized data formats and ARD. This reduces compatibility issues between datasets and tools, enabling smoother integration across different sources and systems. EO Data Cubes also support collaborative development and governance. Because they are often community-driven, multiple stakeholders can contribute to their development, documentation, and evolution. This contrasts with centrally governed infrastructures, where decision-making processes and development roadmaps are typically controlled by a single entity and may evolve according to institutional priorities. Finally, in terms of cost transparency and accessibility, EO Data Cubes can reduce reliance on usage-based pricing models common in commercial cloud platforms. By enabling deployment on

local or hybrid infrastructure, they help avoid hidden costs related to data storage, processing, and egress. Additionally, they often rely on well-documented, openly accessible datasets, improving usability, and lowering barriers for new users.

Overall, EO Data Cubes provide an open, flexible, and transparent alternative to many existing cloud-based EO platforms. They retain the benefits of scalable data access and processing while addressing critical issues related to interoperability, reproducibility, governance, and long-term sustainability. Open Data Cube
In this context, the ODC initiative has emerged as a leading technology for the implementation of EO Data Cubes. The ODC is a free open-source management and analysis architecture solution for handling big geospatial data, including Big Earth Observation Data. It utilizes a PostgreSQL database for metadata indexing and a range of embedded tools and functionalities that allows for efficient spatio-temporal querying, visualization, and analysis of indexed (i.e., catalogued) data and can be easily deployed and run on either a local machine or server. The scalable architecture of the ODC supports parallel processing and distributed computing, making it ideal for handling complex geospatial analyses. Its modular design allows for customization and integration with other tools and libraries, enabling users to build tailored solutions for their specific requirements.

In the ODC, data are organized through a clear separation between physical storage and logical indexing, which enables both flexibility and efficient access. The geospatial data are stored as files in standard formats (e.g., GeoTIFF or COG) on disk, cloud object storage, or other file systems. These files are not moved into a central database; instead, their locations and associated metadata are recorded in an index, usually implemented using PostgreSQL. This index contains detailed information about each dataset, including spatial extent, temporal coverage, spectral bands, and product definitions. Through this metadata-driven approach, ODC can rapidly query and assemble data across space and time without duplicating the underlying files. When a user requests data, the system uses the index to identify relevant datasets and dynamically loads them into analysis environments as structured data arrays. This design ensures scalability, supports distributed storage systems, and allows efficient handling of large EO archives while maintaining consistency and reproducibility.

Through JupyterHub, multiple users can access shared computational resources via web-based notebooks, typically powered by Jupyter Notebook. The user interface and workspace of the ODC are designed to provide a flexible, interactive, and analysis-oriented environment for working with EO data. Within their workspace, users can interact with the ODC, and all the datasets available in it, through Python APIs, enabling them to query datasets by spatial extent, time range, and product type. The workspace is pre-configured with essential geospatial and scientific libraries, allowing users to perform data processing, visualization, and analysis in an integrated environment. It also supports organized project structures where notebooks, scripts, and outputs can be stored and shared, facilitating reproducibility and collaboration. Additionally, users can customize their workspace by installing new libraries, integrating external tools, or developing reusable functions, making the environment highly adaptable to different applications and user needs.

Overall, the ODC provides a powerful and flexible framework for managing and analyzing EO and other geospatial data.

4.3.3 CUBE IN A BOX

One of its main limitations of the ODC lies in the complexity of deployment and configuration. Setting up an ODC instance typically requires significant and diverse technical expertise, including configuring databases (e.g., PostgreSQL), managing storage systems, and orchestrating multiple software components. The Cube-in-a-Box (CiaB) approach directly addresses this limitation by providing a pre-configured, ready-to-deploy version of ODC. Using Docker, the CiaB allows the entire ODC environment, including the PostgreSQL database, data storage configuration, Python libraries, and supporting services, to be packaged into isolated containers. These containers ensure that the software behaves consistently across different machines and operating systems, eliminating many of the dependency and configuration issues that can arise during a traditional installation. By providing a standardized, self-contained solution, CiaB allows to easily deploy a fully functional data cube on local machines, institutional servers, or cloud platforms.

The CiaB approach not only speeds up adoption but also ensures reproducibility and portability of the ODC environment. At the same time, it allows for customization and the integration of local datasets and workflows, maintaining the inherent flexibility and extensibility of the ODC framework while significantly reducing the complexity of traditional deployments. This capability is particularly important for the deployment of the DCs for NOSTRADAMUS pilot areas, which require a core, standardized architecture and shared functionalities, yet must remain adaptable to meet diverse regional needs and requirements.

4.3.4 ODC EXPLORER WEB UI

The ODC Explorer is a web-based UI designed to support the discovery and inspection of EO and other geospatial datasets indexed in the data cube's PostgreSQL/PostGIS database. It provides users with an intuitive and interactive means of accessing the contents of the data cube without requiring direct interaction with backend services or APIs.
Within the LE CiaB environment, the ODC Explorer is deployed alongside the data cube infrastructure. It focuses primarily on metadata exploration, rather than data processing, making it a valuable tool for understanding data availability, coverage, and structure before performing analysis.
Key functionalities of the ODC Explorer include:

- **Dataset discovery**: Users can search and filter datasets by product type, temporal range, and spatial extent.
- **Metadata inspection**: Detailed metadata records are accessible for each dataset, including acquisition time, platform, processing level, and other relevant attributes.
- **Spatial visualization**: An interactive map interface allows users to view dataset footprints and coverage across the region of interest.

- **Temporal navigation**: Time-series capabilities enable users to explore data availability over time.

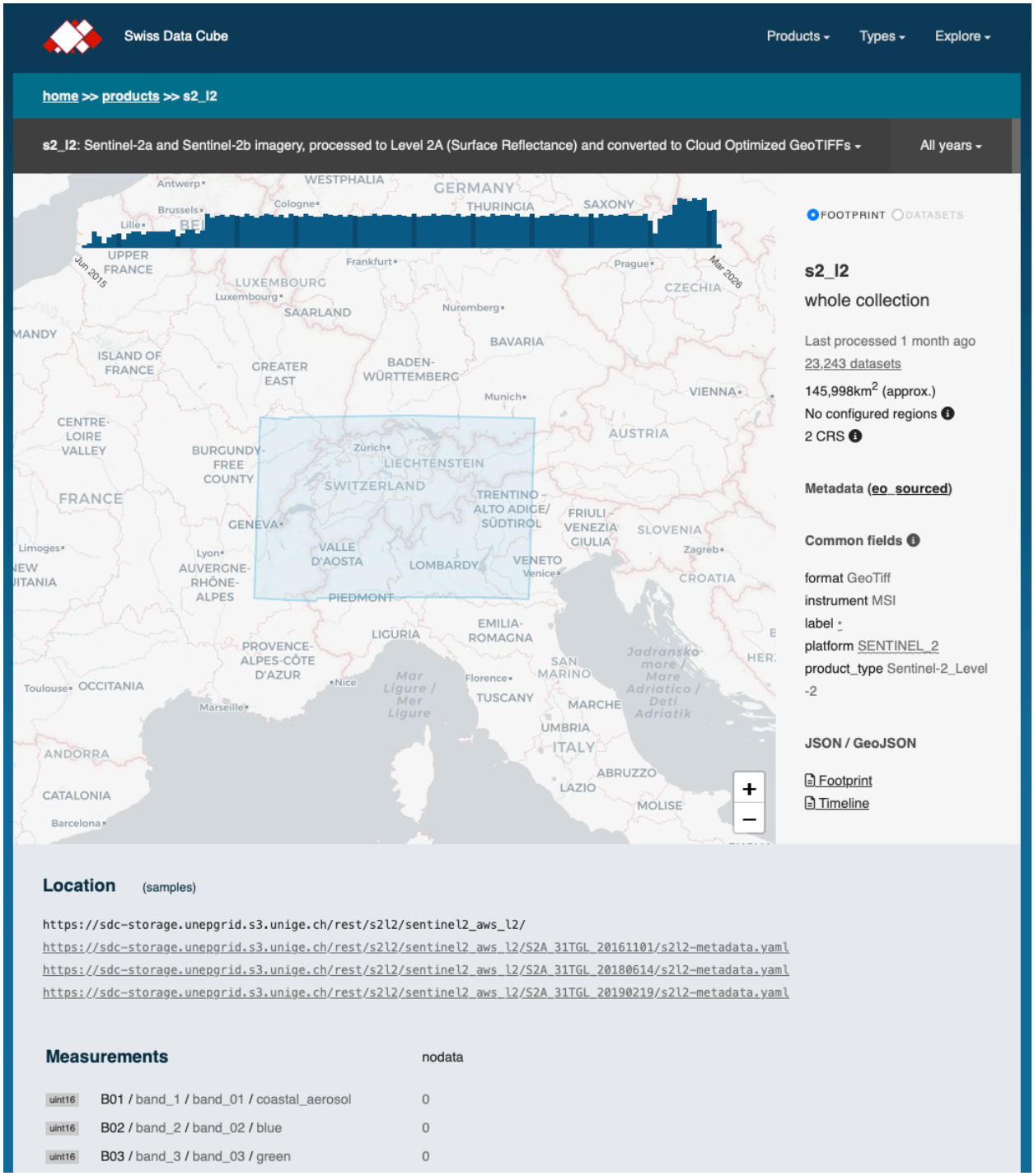

FIGURE 18: ODC EXPLORER OF THE SWISS DATA CUBE

4.3.5 STAC INTERFACE

The SpatioTemporal Asset Catalog (STAC) instance of the Open Data Cube Explorer is a standards-compliant interface that exposes geospatial datasets indexed within the ODC as discoverable and queryable STAC Items, Collections, and Catalogs. By implementing the STAC specification, the ODC Explorer allows users and client applications to search, filter, and access Earth observation data using widely adopted open standards, making it interoperable with a broad ecosystem of STAC-compatible tools such as STAC browsers, QGIS plugins, and Python libraries like pystac and pystac-client. The STAC endpoint typically sits alongside the Explorer's human-facing web interface and provides a machine-readable API, often conforming to the STAC API specification, through which users can perform spatial and temporal queries to retrieve metadata and asset links for datasets like Landsat, Sentinel, or other satellite imagery collections managed within the cube. This integration bridges the gap between the ODC's powerful data management and analysis capabilities and the broader geospatial community's need for standardized data discovery and access, facilitating workflows that range from simple data browsing to automated large-scale processing pipelines.

The implementation details of DCs in NOSTRADAMUS will be further elaborated in Deliverables D5.1-5.3-5.5.

4.4 LOW-CODE APPLICATION PLATFORM

The Low-code Application Platform (LCP) is a crucial component within the NOSTRADAMUS Cloud Platform, designed to democratize the development of data-driven digital solutions for the agricultural sector. It is positioned as the *Democratization layer*, bridging the gap between the technical infrastructure and domain experts.

The NOSTRADAMUS initiative leverages digital technologies to enhance food security, sustainability, and agricultural resilience in the EU, aligning with key EU frameworks such as the European Green Deal and the Common Agricultural Policy. Developing complex cyber-physical applications, which integrate various data sources like EO data, real-time IoT streams, Data Cubes, and cloud-native analytics, can be technically challenging for IT experts. The LCP addresses this challenge by providing tools that enable experts, including SMEs, Agri-industry, Cooperatives, Agri-food companies, to build custom digital solutions, by offloading technical and technological complexity barriers.

The LCP is one of the three distinct but highly integrated functional pillars of the NOSTRADAMUS cloud-native infrastructure, alongside the Data Infrastructure and the IoT Observatory. It is built upon a hybrid Edge-to-Cloud architecture. The LCP's development is a specific deliverable of Work Package 2 (WP2), with the Aristotle University of Thessaloniki (AUTH) leading Task 2.4, "Develop generation engine for digital applications".

The LCP development is also closely integrated with other deliverables and work packages:

- It serves as the foundation for Deliverable D2.5 (Application Generation Engine) by defining the structural rules, component interactions, and "Edge-to-Cloud" logic that the engine will automate.
- It transforms user-centric data needs identified in D2.1 into technical specifications for the NOSTRADAMUS architecture.
- The generated applications are central to Work Package 6 (WP6), which focuses on third-party funding for digital applications and capacity building, ensuring interoperability with the broader NOSTRADAMUS ecosystem.
- AUTH, as the lead beneficiary for Task 2.4, is responsible for developing the bootstrapping software for the Application Generation Engine.
- AUTH also contributes to training IT experts and providers on how to use this bootstrapping software to develop open-source digital applications. These training activities include online workshops and recorded demonstrations (planned for months 33-35 of the project).
- The LCP facilitates the creation of GUI components that interact with Data Cubes from WP5.

A key milestone associated with the LCP is the "First version of App. Generation Engine" (MS4), due at month 30 of the project, verifying that the engine can be used to create applications. The second version of the Application Generation Engine (D2.6) is due at month 44.

4.4.1 CORE FUNCTIONALITY AND APPROACH

The Nostradamus LCP employs a Model-Driven Engineering (MDE) approach, utilizing Domain-Specific Languages (DSLs) to enable experts to design and automatically generate and deploy software for different parts of their applications. This approach allows for a "citizen developer" concept, where non-technical end-users can auto-generate parts or the entirety of their applications by simply designing the solution rather than programming it. This process is intended to make real-world application development easy, fast, and error-free, significantly reducing the technical burden and latency traditionally associated with programming complex cyber-physical systems.

A key innovation driving this approach is an AI assistant, supported by a multi-agent backend, which acts as a natural language interface. This assistant empowers domain experts, such as farmers, agronomists, and policymakers, to describe their desired monitoring workflows or application requirements using plain text. The AI assistant's crucial role is to then translate these natural language descriptions into formal Domain-Specific Language (DSL) models. These DSL models encapsulate the specific functionalities and behaviors of the desired application without requiring the user to write complex code. This process is effectively an intermediary AI engineering step, consolidating domain expert knowledge into a structured, machine-readable format.

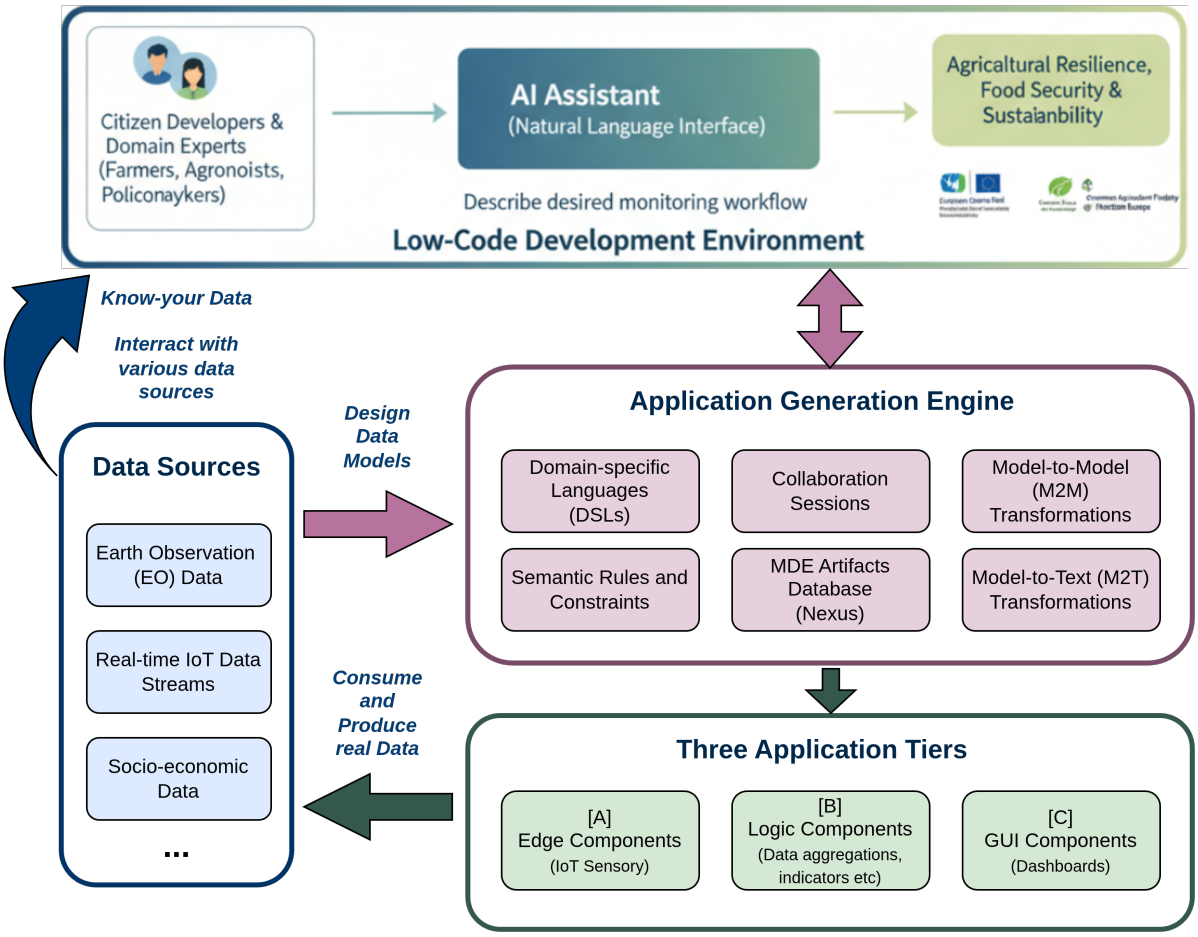

FIGURE 19: LOW-CODE CONCEPTUAL FRAMEWORK

Once these DSL models are created (either through the AI assistant or directly by users with more technical proficiency), they are fed into the Application Generation Engine. The AGE is the core component responsible for transforming these high-level designs and DSL models into executable software artifacts. It achieves this automation by employing Model-to-Model (M2M) and Model-to-Text (M2T) transformations. This engine, which is a specific deliverable of Work Package 2 (D2.3 - Application Generation Engine) led by AUTH, ensures that the generated applications adhere to predefined architectural patterns, security standards, and interoperability requirements through embedded semantic rules and constraints.

The Nostradamus Low-Code Platform primarily benefits two categories of user:

- **Domain Experts / Citizen Developers**: These users, such as farmers, agronomists, and policymakers, are the primary beneficiaries of the AI assistant. Their interaction primarily involves describing their needs and workflows using natural language. The AI assistant then translates these into DSL models, abstracting much of the technical complexity. These users are then enabled to leverage these generated DSL models through the Application

Generation Engine to create and deploy their custom digital solutions. This feature of the LCP is experimental and is active research of ISSEL (AUTH).

- **IT Experts / Developers**: While the AI assistant democratizes access, more IT users may also engage with the platform. They might either use the AI assistant for initial model scaffolding or directly interact with and refine the generated DSL models. These users can also directly utilize the Application Generation Engine with custom or refined DSL models for more advanced customization, integration, or to develop complex applications, thus benefiting from the efficiency and structure of the model-driven approach while retaining granular control.

This model-driven approach defines the structural rules and "Edge-to-Cloud" logic that will be automated by the "D2-3 - Application Generation Engine". It supports core modules such as drought monitoring, crop suitability, soil fertility, and sustainable pest management. Furthermore, it serves as the foundation for third parties to develop interoperable digital applications through Open Calls (WP6), fostering innovation within the agricultural ecosystem. This paradigm shift significantly reduces the technical burden and latency traditionally associated with programming complex cyber-physical systems, ensuring that digital solutions are secure, interoperable, and precisely tailored to the agricultural sector's unique socio-economic and biogeographical conditions.

4.4.2 MODEL-DRIVEN ENGINEERING AND DOMAIN-SPECIFIC LANGUAGES

At the heart of the LCP is the use of specialized meta-models and Domain-Specific Languages (DSLs), both graphical and textual, that are tailored to the agricultural domain. Unlike general-purpose programming languages (GPLs), such as Python or Java, these DSLs allow experts to describe specific system characteristics, such as:

- **Hardware and Network Configurations:** Describing edge device properties (e.g., Raspberry Pi, ESP32), sensor types, and communication protocols (MQTT, CoAP).
- **Analytical Logic:** Defining data aggregation flows, filtering criteria, and the generation of food-security indicators.
- **Visual Requirements:** Specifying dashboard elements and data stream monitoring parameters. By shifting the focus from writing code to defining models, the platform ensures the reusability of models across different demonstration sites and projects.

4.4.3 APPLICATION GENERATION ENGINE

The Application Generation Engine (AGE) is the core component responsible for transforming high-level designs and DSL models into executable software artifacts. It employs Model-to-Model (M2M) and Model-to-Text (M2T) transformations to achieve this automation. This engine, a specific deliverable of Work Package 2 (D2.5 and D2.6) led by AUTH, relies on several key elements to function:

- **Domain-specific Languages (DSLs)**: These are tailored languages that allow experts to describe specific aspects of their application, such as edge device characteristics, sensor

configurations, visualization requirements, and logical data flows, without needing general-purpose programming expertise.

- **Semantic Rules and Constraints**: These embedded rules ensure that the generated applications adhere to predefined architectural patterns, security standards, and interoperability requirements, ensuring the quality and consistency of the output.
- **Collaboration Sessions**: The diagram indicates that the AGE supports collaboration sessions, suggesting that multiple users or stakeholders can contribute to the design process.
- **MDE Artifacts Database** This database likely stores and manages the various models, transformations, and generated code components, acting as a central repository for the MDE process.

AGE is specifically tasked with creating software components for the **three application tiers** of the cyber-physical application model:

- **Edge Components**: the user may be able to describe their edge devices characteristics, as well as the sensors they are equipped with, so as to automatically generate software for the embedded devices that will handle the dispatch of data to the cloud.
- **App Logic Components**: automatic generation of data aggregation software will occur for the App Logic components, and graphical-based languages will be developed (or used) towards describing logic flows of data or processes.
- **App GUI Components**: the user may describe which data wants visualized and in what format, to automatically generate the visual parts of the App GUI components, as well as the necessary software to achieve connectivity to the other parts of the infrastructure

4.4.4 AI ASSISTED DEVELOPMENT

A critical innovation within the Nostradamus Low-Code Platform (III) is the integration of an **AI Assistant** supported by a **multi-agentic backend**. This component is designed to fundamentally lower the entry barrier for agricultural domain experts—such as farmers, agronomists, and policymakers—by allowing them to participate in the software creation process through high-level abstractions rather than manual coding.

The platform introduces a "Vibe Coding" paradigm, where the traditional requirement of mastering complex programming syntax is replaced by a natural language interface. Under this model, a domain expert can describe a desired monitoring workflow or a specific agricultural logic in plain text (e.g., *"Alert me if the soil moisture in the vineyard stays below 15% for more than three days during the ripening stage"*).

The AI-driven development workflow operates through several sophisticated stages:

- **Natural Language Translation:** Utilizing Large Language Models (LLMs), the AI assistant interprets the user's verbal or textual requirements and translates them into formal, syntactically correct Domain-Specific Language (DSL) models.

- **Automated Model Synthesis:** These high-level models are then processed by the Application Generation Engine, which performs automated Model-to-Model and Model-to-Text transformations. This process synthesizes the necessary source code for the three application tiers: Edge Components for sensory acquisition, Logic Components for cloud-deployed data stream pipelining, and GUI Components for interactive dashboards.
- **Syntactic Validation and Reasoning:** To ensure that the generated applications are robust and secure, the AI backend performs syntactic validation and reasoning. This ensures that the generated models adhere to predefined architectural patterns and semantic constraints, preventing the creation of error-prone or insecure software artifacts.
- **Reducing Development Latency:** By bypassing the need for General-Purpose Languages (GPLs) and manual implementation, this AI-assisted approach significantly reduces development latency and the deep technical burden traditionally associated with programming complex cyber-physical systems.

This democratization of technology ensures that digital solutions are precisely tailored to the unique socio-economic and biogeographical conditions of the agricultural sector, empowering stakeholders to optimize resource efficiency and resilience without requiring a strong technological background.

5 DATA PIPELINES

The NOSTRADAMUS platform implements a set of coordinated data pipelines supporting the ingestion, transformation, integration, and delivery of heterogeneous data sources. These pipelines operationalize the architecture described in the previous sections and provide the mechanisms through which EO, IoT, and socio-economic data are converted into harmonized and exploitable information products. The pipelines enable the generation of standardized datasets that are consumed by downstream services, including Data Cubes, analytical modules, dashboards, and digital applications.

The pipeline architecture is designed to support different processing paradigms according to the characteristics of the data being handled. EO data pipelines are primarily batch-oriented and focus on the acquisition, transformation, and harmonization of large-volume datasets. IoT pipelines are oriented toward near real-time ingestion and stream processing, supporting continuous monitoring and rapid event handling. Socio-economic pipelines focus on the structured integration of statistical and contextual datasets that complement EO and IoT observations. Together, these pipelines form the operational backbone of the NOSTRADAMUS platform and support the full chain from data acquisition to analytical exploitation.

To ensure interoperability and long-term reusability, the pipelines rely on standardised metadata, interoperable data formats, and well-defined integration interfaces. At the platform level, they converge into a common data provisioning layer that feeds the Data Cube infrastructure and supports application-specific workflows.

5.1 OVERVIEW OF DATA PIPELINES

The data pipeline architecture of NOSTRADAMUS is organized around domain-specific but interoperable processing chains. Each pipeline is responsible for acquiring data from one or more upstream sources, applying validation and transformation steps, and exposing the resulting datasets to downstream components through standard interfaces. Although each pipeline serves different data types and operational needs, they all follow a common architectural logic based on traceability, modularity, and reproducibility.

At a high level, the pipelines perform four main functions:

1. **Data acquisition**, including retrieval from external repositories, APIs, sensors, or partner-maintained sources
2. **Data transformation and preprocessing**, including harmonization, quality control, and conversion into analysis-ready formats
3. **Metadata registration and discoverability**, ensuring that datasets can be indexed, searched, and referenced consistently
4. **Data provisioning**, enabling access by Data Cubes, analytical modules, notebooks, dashboards, and other application services

This architecture supports both batch and near real-time execution. Large EO datasets are typically processed through declarative, workflow-driven pipelines, while IoT streams are handled through event-driven mechanisms. Socio-economic data pipelines provide slower moving but semantically rich datasets that support contextualization, policy analysis, and integration with the broader data ecosystem.

Figure 20 conceptually illustrates the NOSTRADAMUS data pipeline architecture, organized as three parallel, domain-specific processing streams. Each stream represents a core data type: the blue *Batch Layer* for EO data, the green *Speed Layer* for high-velocity IoT data, and the orange *Contextual Layer* for socio-economic data. The diagram shows how each pipeline follows a common logical flow from left to right, beginning with data acquisition from heterogeneous sources, followed by domain-specific transformation and pre-processing steps. A key feature of the architecture is the central convergence at the metadata registration stage, where all pipelines contribute to a unified, FAIR-compliant STAC catalog, ensuring interoperability and discoverability.

Finally, the processed and cataloged data from all three streams is provisioned to a common set of downstream consumers, including the Data Cubes, core agricultural modules, and interactive analysis environments, highlighting the platform's integrated approach to multi-source data fusion.

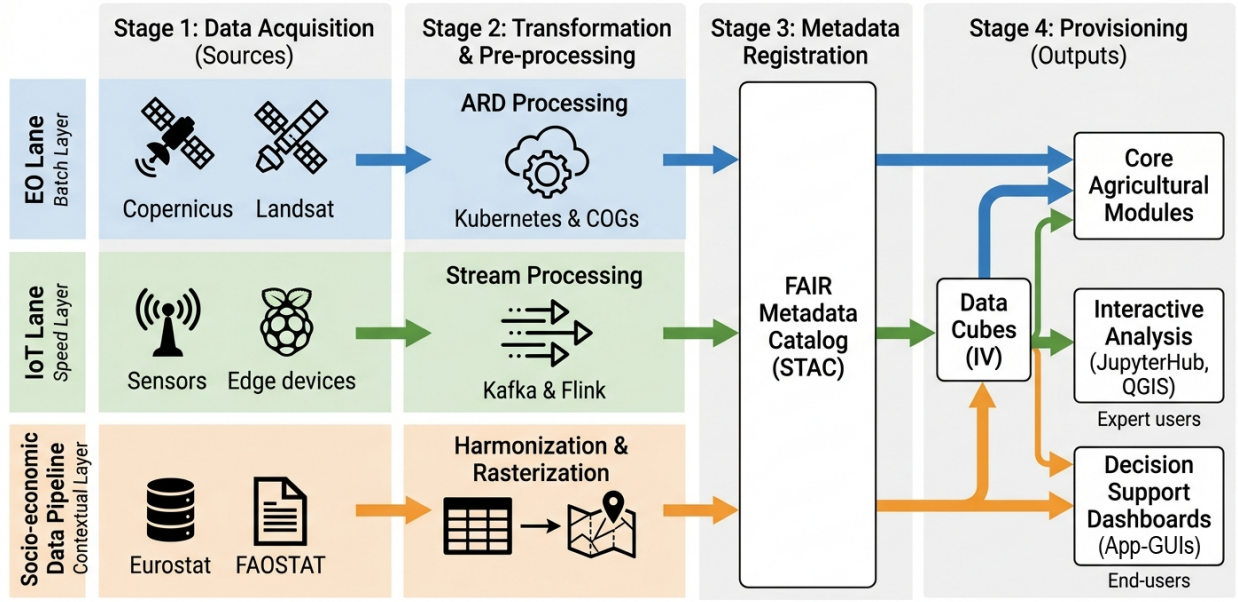

FIGURE 20: NOSTRADAMUS DATA PIPELINE CONCEPTUAL ARCHITECTURE

The coordination of these pipelines ensures that NOSTRADAMUS can combine static and dynamic data sources within a single platform architecture. This is essential for supporting the project's hybrid cloud-edge model and for enabling cross-domain applications that require EO, IoT, and socio-economic information in a consistent and interoperable way.

5.2 EO DATA PIPELINES

The EO Data Pipelines, implemented within the Data Hub (I), provide the core infrastructure for automated ingestion, large-scale processing, and standardized provisioning of EO data. These pipelines operationalize the *Batch Layer* of the NOSTRADAMUS architecture, transforming raw satellite measurements into ARD that can be directly exploited by scientific workflows, core agricultural modules, and the Data Cube ecosystem.

By coordinating these pipelines, the Data Hub acts as a consolidated provisioning environment that transforms raw satellite imagery into the standardized, actionable agricultural intelligence required for European food security.

5.2.1 EO DATA LIFECYCLE

The primary objective of these pipelines is to establish a reliable and user-friendly service that manages the full EO data lifecycle. Data is acquired from diverse external repositories to ensure a comprehensive view of European agricultural landscapes:

- **Copernicus Sentinel Missions:** Ingestion of Sentinel-1 (radar) and Sentinel-2 (optical) imagery for real-time monitoring.

- **NASA/USGS Missions:** Integration of historical and current Landsat archives (Landsat 5, 7, 8, 9) to support long-term trend analysis.
- **Third-Party and In-Situ Data:** Acquisition of ancillary datasets and in-situ ground measurements to support atmospheric correction, validation, and multi-sensor harmonization.
- **Commercial Datasets:** Provision for high-resolution commercial satellite data through CSCDA agreements where required by specific pilot use cases.

5.2.2 DATA INGESTION AND PROCESSING WORKFLOWS

The ingestion and processing workflows are implemented as Kubernetes-orchestrated pipelines that execute the following automated steps:

1. **Calibration & Pre-processing:** Applying sensor-specific corrections, orthorectification, and atmospheric correction to convert raw products into harmonized ARD.
2. **Format Harmonization:** Converting raster data into Cloud Optimized GeoTIFFs. This format enables HTTP range requests, allowing applications to retrieve only the specific byte-ranges required for a task without downloading massive files, thereby reducing latency and computational overhead.
3. **Tiling and Reprojection:** Aligning data with the common grid structures and spatial reference systems required by the Data Cubes (IV) to support multi-dimensional spatiotemporal analysis.

5.2.3 METADATA MANAGEMENT AND DISCOVERY

A critical characteristic of the EO pipelines is the systematic extraction and registration of metadata to ensure the platform remains fully adherent to the FAIR Principles.

- **STAC Catalog:** All EO assets are indexed using the STAC standard, which decouples JSON-based metadata from the actual binary data assets.
- **Granular Discovery:** The STAC API enables users and automated services to perform complex real-time searches based on spatial bounding boxes, temporal intervals, cloud cover percentage, or specific satellite missions.
- **Traceability:** Every asset in the catalog preserves its processing lineage, providing full traceability from the initial satellite downlink to the final indicator produced for the end-user.

5.2.4 EXPOSURE AND PROVISIONING TO DOWNSTREAM COMPONENTS

Processed EO products are exposed through standardized interfaces to support the broader NOSTRADAMUS ecosystem:

- **Data Cube Integration:** ARD is pipelined into the Data Cubes (IV), where it is indexed for high-performance "pixel-drilling" and multi-dimensional time-series analysis.

- **Core Module Support:** The pipelines provide high-veracity inputs for the four core agricultural modules: drought monitoring, crop suitability, soil fertility, and pest management.
- **Interactive Analysis:** Expert users can access datasets through containerized environments like JupyterHub, utilizing Python APIs, VS Code, or QGIS for user-driven scientific research.
- **Secure Access:** Access to assets in object storage (MinIO S3) is tightly controlled via Pre-Signed URLs, granting time-bound permissions for authorized human or machine-to-machine data retrieval.

These EO pipelines are operational within the Data Hub environment and constitute the primary mechanism for EO data provisioning across the NOSTRADAMUS platform.

5.3 IOT DATA PIPELINES

The IoT Data Pipelines constitute the "Speed Layer" of the NOSTRADAMUS digital ecosystem, specifically engineered to manage high-velocity, event-based data originating from the physical agricultural environment. These pipelines operationalize the IoT Observatory (II), transforming raw sensor readings into actionable intelligence through a hybrid Edge-to-Cloud architecture that reconciles the need for immediate real-time responses with long-term historical accuracy.

5.3.1 INGESTION OF SENSOR AND EDGE-DEVICE DATA

The primary entry point for all field-level telemetry is the Edge Gateway (Figure 2 - part of IoTO), a robust abstraction layer designed to handle the heterogeneous and often intermittent nature of agricultural connectivity.

- **Pilot Site Integration:** The pipelines ingest high-resolution environmental data from the five strategic demonstration sites (Cyprus, Germany, Switzerland, Slovenia, and Serbia). These streams include parameters such as soil moisture, microclimate variables, atmospheric pressure, and leaf wetness.
- **Multi-Protocol Support:** To ensure interoperability with diverse hardware (e.g., Raspberry Pi, ESP32), the gateway exposes multiple high-level interfaces: a REST API for standard request-response submission, MQTT for optimized edge communication, and CoAP for lightweight, low-bandwidth transmission in remote fields with weak connectivity.
- **Automated Validation:** Incoming events undergo semi-automated schema identification to ensure they conform to the expected formats before being published to the message broker.

5.3.2 STREAM PROCESSING AND BUFFERING MECHANISMS

Once ingested, data is buffered and decoupled through an Apache Kafka cluster, which acts as the system's data stream middleware.

- **Persistent Buffering:** Kafka ensures that sudden spikes in data ingestion—common during extreme weather events—do not overwhelm downstream analytics. Data is organized into a strict hierarchy of Organizations, Projects, and Collections, with Kafka topics following the naming convention: {organization}.{project}.{collection}.
- **Real-time Analytics:** The pipeline utilizes two complementary engines to transform raw streams:
    - ksqlDB: Enables SQL-like querying on live streams for immediate filtering and standard transformations.
    - Apache Flink: Handles Complex Event Processing (CEP) and integrates Machine Learning pipelines directly into the stream, enabling real-time inference (e.g., pest detection or frost alerts).

5.3.3 INTERFACES WITH EDGE SYSTEMS AND CLOUD SERVICES

The pipeline serves as the vital link between the physical environment and the cloud's analytical capabilities:

- **Edge-to-Cloud Continuum:** The Edge Gateway facilitates a two-way flow, where edge components dispatch data to the cloud and can receive commands or configuration updates back from the monitoring services.
- **Unified Serving Layer:** This layer merges the real-time views with historical batch data, abstracting the origin of the data from the end-user. It exposes all functionalities via a standardized REST API (OpenAPI v3) and a Python SDK, allowing third-party developers to rapidly integrate IoT data into their agricultural solutions.

5.3.4 EXPOSURE TO DOWNSTREAM ANALYTICAL COMPONENTS

The processed IoT insights are seamlessly integrated with the broader NOSTRADAMUS ecosystem:

- **Multi-source Fusion:** IoT streams are combined with Earth Observation (EO) Analysis Ready Data from the Data Hub (I) and socio-economic datasets.
- **Data Cube Ingestion:** Rasterized IoT data can be pipelined into the Data Cubes (IV), enabling multi-dimensional spatiotemporal analysis such as "pixel-drilling" across both satellite imagery and ground-sensor time-series.
- **Decision Support:** The final indicators are delivered to App-GUI Components (C) for visualization in interactive dashboards, empowering farmers and policymakers with a unified view of field-level events and macro-level trends.

5.4 SOCIO-ECONOMIC DATA PIPELINES

The Socio-economic Data Pipelines are designed to provide the contextual and statistical foundation necessary to transform physical Earth Observation (EO) and IoT measurements into actionable policy and business intelligence. While the IoT Observatory (II) handles high-velocity

events and the Data Infrastructure (I) manages massive raster archives, these pipelines focus on semantically rich datasets that quantify the human and economic dimensions of European food security.

5.4.1 DATA GAP ANALYSIS

The primary objective of these pipelines, operationalized through Task 3.3, is to identify and address the current data GAP in local and regional agricultural statistics. Current data management in agriculture often suffers from scattered data and a lack of interoperable solutions for capturing research-driven socio-economic insights. To bridge this gap, the pipeline ingests data from diverse, high-veracity sources:

- **International & EU Repositories:** Eurostat, the World Bank (WDI), FAOSTAT, and the Farm Accountancy Data Network (FADN).
- **National & Regional Sources:** National Statistical Offices and specialized market reports (e.g., Euromonitor).
- **Demonstration Site Contexts:** Socio-economic profiles from the five European pilot sites (CY, DE, CH, RS, SL) to ensure the data is tailored to local biogeographical conditions.

5.4.2 CATEGORIZATION OF SOCIO-ECONOMIC DATA PRODUCTS

To support the digital applications being developed by third parties in WP6, the pipeline organizes data into two specialized groups of solutions:

- **Group A - Farming Business Datasets:** This set focuses on the micro-level economic performance of agriculture. It includes variables such as crop prices, production, trade patterns, machinery performance, and market demand. These datasets enable the development of "App - Logic Components (B)" for farm-level decision-making and yield optimization.
- **Group B - Rural Community & Well-being Datasets:** This group focuses on the macro-level indicators necessary for policy evaluation. Key variables include farmer income, poverty levels, educational attainment, and demographic trends. These indicators are essential for the European Green Deal and CAP monitoring, reporting, and verification (MRV) frameworks.

5.4.3 INTEGRATION AND HARMONIZATION

Unlike the real-time stream processing of IoT data, these pipelines are primarily batch-oriented and focus on the structured integration of heterogeneous statistical formats:

- **Acquisition & Pre-processing:** Data is retrieved via APIs or structured files (CSV, XML, Statistics formats like SAS/DTA) and stored in private, temporary repositories when confidentiality is required.

- **Transformation (Rasterization):** A key innovation of the NOSTRADAMUS approach is the rasterization of statistical data. By converting tabular socio-economic indicators into georeferenced layers, the platform allows these datasets to be ingested directly into the Data Cubes, enabling multi-dimensional analysis alongside satellite imagery and soil moisture maps.
- **Metadata Registration:** Every socio-economic asset is documented using the STAC-compliant catalogue to ensure it adheres to the FAIR Principles (Findable, Accessible, Interoperable, Reusable).

5.4.4 IMPACT ON POLICY AND DECISION SUPPORT

By combining these pipelines with EO and IoT data, the NOSTRADAMUS architecture enables a holistic evaluation of the agricultural sector. These pipelines specifically support the project's goal to improve the energy balance of data-based solutions and enhance the deployment of digital tools in areas with weak connectivity, such as the Cyprus demonstration site. Ultimately, the socio-economic data provides the "Expert Information" required for stakeholders to make informed decisions about resource efficiency, agrochemical reduction, and long-term climate adaptation strategies.

5.5 INTEGRATION AND DATA FLOW ACROSS PIPELINES

Although the platform includes multiple domain-specific pipelines, the value of the NOSTRADAMUS architecture lies in their coordinated integration. EO, IoT, and socio-economic pipelines are not isolated processing chains; instead, they contribute complementary information to a common data ecosystem that supports the project's analytical and application layers.

Integration across pipelines is achieved through shared architectural principles and standardised interfaces. These include the use of harmonized metadata models, structured dataset registration, interoperable APIs, and common processing and orchestration practices. Through this approach, datasets originating from different sources and processing paradigms can be referenced, combined, and consumed within a unified platform environment.

In practice, this integrated model enables multiple types of downstream exploitation. The Data Cube infrastructure can consume harmonized EO data and, where appropriate, combine them with other datasets for multi-dimensional analysis. Application modules can access preprocessed EO products, live or historical IoT streams, and socio-economic indicators according to their analytical needs. Interactive user environments can similarly access these data sources through notebooks, APIs, and other service interfaces.

The role of the Data Hub within this integrated model is to act as the principal provisioning environment for EO data and processing workflows, while maintaining interoperability with other platform components. The integration with IoT and socio-economic pipelines ensures that the final

platform does not simply host independent data silos but provides a coordinated data architecture capable of supporting cross-domain applications and decision-support services.

5.6 OPERATIONAL CHARACTERISTICS

Across the platform, data pipelines are designed with a strong emphasis on reproducibility, scalability, and operational robustness. Processing workflows are expressed in a declarative manner where possible, use containerized execution environments, and are deployed within a cloud-native orchestration framework. This enables workflows to be executed consistently across infrastructures and supports the long-term portability of the processing chains.

From an operational perspective, the orchestration environment supports batch execution, event-driven triggers, scheduled runs, and interactive invocation by users or services. This flexibility is necessary to accommodate the diversity of NOSTRADAMUS processing scenarios, ranging from bulk EO ingestion to near real-time data handling and application-driven processing.

The principal characteristics of the NOSTRADAMUS data pipelines can be summarized as follows:

- **Reproducibility**, through standardized workflow descriptions and containerized execution
- **Scalability**, through cloud-native orchestration and distributed processing
- **Interoperability**, using shared metadata models, standard data formats, and APIs
- **Traceability**, through metadata registration, lineage preservation, and structured data management
- **Flexibility**, by supporting both automated and user-driven execution patterns

These characteristics are essential to ensure that NOSTRADAMUS can support a broad range of use cases, from scientific experimentation to operational decision support, while maintaining consistency across its different data domains and technical subsystems.

6 CLOUD, EDGE, AND MIXED APPLICATIONS

The NOSTRADAMUS project is designed to address critical challenges in food security, sustainability, and agricultural resilience within the European Union. To achieve these objectives, it establishes a robust, data-centric foundation for agricultural independence by leveraging a hybrid Edge-to-Cloud architecture. This section details the comprehensive framework for cloud, edge, and mixed applications within the NOSTRADAMUS digital ecosystem, emphasizing their cyber-physical nature and the role of the Low-Code Platform in their development.

6.1 THE EDGE-TO-CLOUD CONTINUUM

The continuum is designed to reconcile the conflicting requirements of sub-second Velocity for immediate field reactions and high-precision Veracity for long-term policy and farm planning. To achieve this, the architecture strategically distributes computational logic across two primary layers:

- **The Edge Layer (Observation):** Positioned directly in the agricultural field, this layer handles immediate, low-latency reactions and localized data filtering. By executing App-Edge Components on embedded devices (e.g., Raspberry Pi, ESP32), the system can perform primary sensory data acquisition and local actuation logic without waiting for cloud roundtrips. This is critical for time-sensitive tasks, such as frost alerts or precision irrigation triggers.
- **The Cloud Layer (Analytical Intelligence):** This layer serves as the "heavy lifting" engine for the project. It manages the massive volume of Earth Observation (EO) archives, the Data Cubes, and complex socio-economic datasets. It implements a fusion of Lambda and Kappa architectural patterns to maintain an immutable "source of truth" while simultaneously calculating real-time ad-hoc user queries.

Furthermore, a core innovation of the NOSTRADAMUS continuum is its focus on areas with weak connectivity, such as the Cyprus demonstration site. Unlike purely cloud-reliant systems, the NOSTRADAMUS architecture acknowledges that agricultural internet access is often intermittent or low-bandwidth.

- **Offline Capability:** By emphasizing edge computing, the framework allows App-Edge Components to continue functioning independently when connectivity is lost.
- **Data Synchronization:** Once a connection is established, the Edge Gateway acts as a robust abstraction layer, utilizing lightweight protocols like MQTT and CoAP to synchronize buffered data with the IoT Observatory.

The continuum facilitates a seamless two-way flow of information that empowers stakeholders across the food supply chain:

1. **Sensing:** Field events are captured by Edge Components and dispatched to the cloud.
2. **Harmonization:** The IoT Observatory ingests these streams, fusing them with satellite-derived Analysis Ready Data (ARD) from the Data Hub.
3. **Analysis:** App-Logic Components execute domain-specific machine learning models to generate high-level food security indicators (e.g., crop suitability or drought indices).

4. **Actionable Insight:** The results are delivered via GUI Components in interactive dashboards, allowing farmers and policymakers to optimize operations, reduce agrochemical use, and improve the overall energy balance of data-driven agriculture.

By orchestrating resources across this continuum, NOSTRADAMUS provides a scalable, open-source pattern for the technological development of EU agriculture, ensuring that advanced digital tools are inclusive, robust, and capable of operating under diverse biogeographical and socio-economic conditions.

6.2 THREE-TIER APPLICATION ARCHITECTURE

The NOSTRADAMUS architecture is built upon a hybrid Edge-to-Cloud model and functions as a System-of-Systems, specifically engineered to tackle data fragmentation challenges in European agriculture. It effectively manages the "four V's" of agricultural Big Data the massive Volume of Earth Observation (EO) archives, the high Variety of heterogeneous IoT and socio-economic data, the sub-second Velocity needed for near real-time decision-making, and the high Veracity required for reliable decision support.

This model strategically distributes processing capabilities, utilizing the Cloud Layer for extensive processing and long-term data storage, while the Edge Layer handles immediate, low-latency reactions and localized data filtering. The system is built with a separation of concerns strategy, dividing the cloud-native infrastructure into three distinct, yet highly integrated functional pillars: the Data Infrastructure (I), the IoT Observatory (II), and the Low-Code Platform (III).

Digital solutions developed within the NOSTRADAMUS ecosystem are defined as cyber-physical applications. These applications orchestrate resources across the entire field-to-cloud continuum and are categorized into three functional components or tiers to streamline development, particularly for non-technical domain experts.

6.2.1 EDGE COMPONENTS

Edge Components represent the system's physical interface in the agricultural field. These are software/firmware designed to run on embedded devices, such as Raspberry Pi or ESP32 boards. Their primary functions include acquiring sensory data and executing local actuation logic. These components connect to the NOSTRADAMUS Cloud Platform via an IoT Broker to dispatch collected data to the IoT Observatory (II). The Edge Gateway acts as the cloud-side interface for these components. The Low-Code Platform (III) facilitates the generation of the necessary embedded software and connectivity code for these devices by allowing users to describe their specific hardware characteristics and sensor configurations. The system also supports concurrent data ingestion from multiple heterogeneous sources (IoT devices, sensors) without performance degradation.

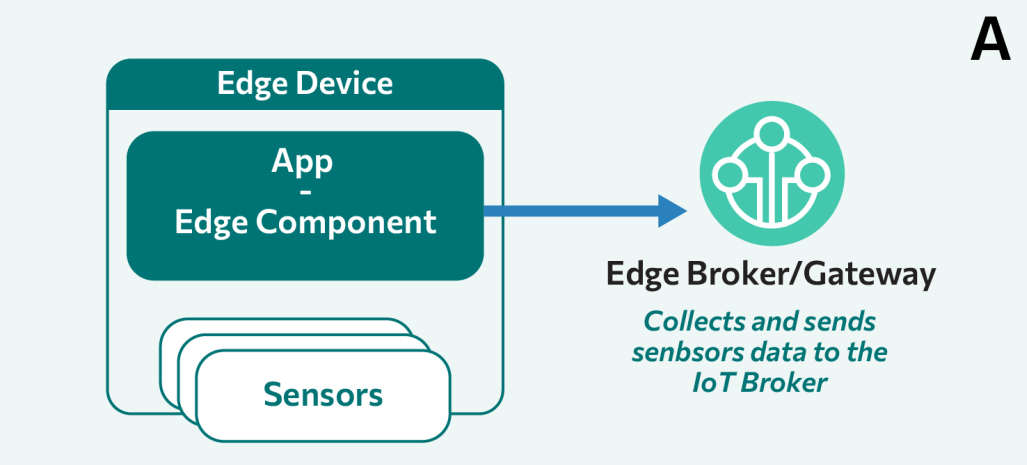

FIGURE 21: APPLICATION TYPE A - EDGE COMPONENTS

This type of applications is responsible for:

- **Sensory Data Acquisition:** Capturing real-time environmental variables such as soil moisture, temperature, and atmospheric pressure.
- **Cloud Dispatch:** Connecting to the NOSTRADAMUS Cloud Platform via the Edge Gateway component of IoTO to send events.

6.2.2 LOGIC COMPONENTS

App-Logic Components serve as the analytical engine for digital applications, residing predominantly within the cloud environment. Their core function involves ingesting data from a variety of heterogeneous sources. These sources include real-time streams originating from App-Edge Components, historical multi-dimensional arrays stored in Data Cubes (IV), and external socio-economic Datasets.

These components are responsible for executing domain-specific algorithms, which encompass techniques such as signal processing and machine learning. Through these algorithms, raw data is transformed into high-level indicators and complex predictions, providing actionable agricultural intelligence. The platform facilitates the automated generation of data aggregation software for these components, employing graphical languages to describe logical data flows and processes without requiring manual programming.

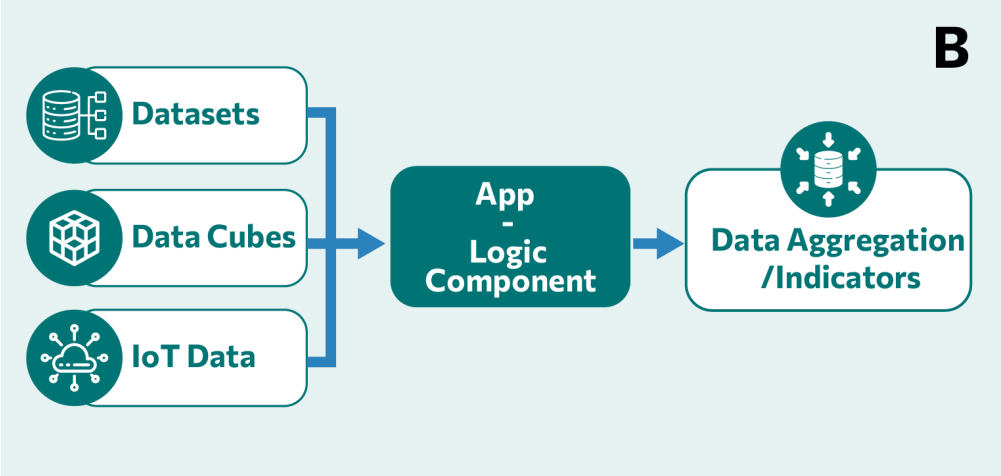

FIGURE 22: APPLICATION TYPE B – LOGIC COMPONENTS

The primary functionality of this type of applications is the ingestion and fusion of heterogeneous data sources, including:

- Real-time streams from App-Edge Components.
- Historical multi-dimensional arrays from Data Cubes.
- External socio-economic Datasets (e.g., crop prices, trade patterns). These components execute domain-specific algorithms—such as signal processing and machine learning—to transform raw data into high-level food security indicators and complex predictions.

6.2.3 GUI COMPONENTS

The GUI Components constitute the user-facing layer of the application, facilitating hybrid data acquisition and visual representation in the form of interactive Dashboards. These components are responsible for visualizing the insights and decisions generated by the Logic Components for end-users, such as farmers and policymakers. Utilizing the Low-Code Platform (III), IT target groups can visually define which data streams they wish to monitor and in what format. This triggers the automatic generation of the visual interface, and the underlying software needed to maintain connectivity with the rest of the infrastructure.

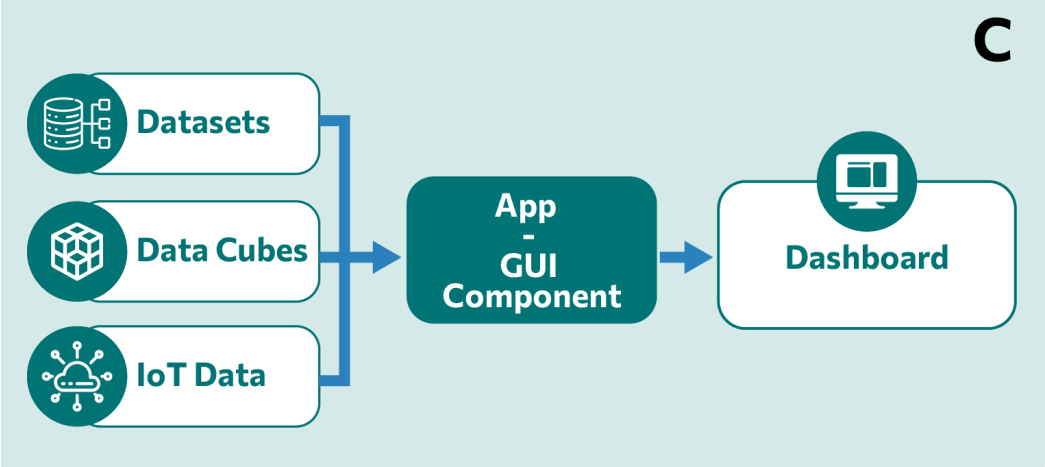

FIGURE 23: APPLICATION TYPE C – GUI COMPONENTS

This type of applications provides the following core functionality:

- **Multi-dimensional Data Visualization:** Presenting complex decisions, indicators, and insights, such as crop suitability indices or drought alerts.
- **Hybrid Data Interaction:** Facilitating the acquisition and inspection of heterogeneous data sources, such as from IoTO, the Data Hub, and Data Cubes.
- **Customizable Stakeholder Dashboards:** Providing a user-friendly environment where different stakeholders can create personalized dashboards by utilizing a palette of predefined visualization components (e.g., maps, charts, and alerts) tailored to their specific regional or operational need.

6.3 INTEROPERABILITY AND SECURITY STANDARDS

To prevent vendor lock-in and ensure wide adoption, applications must adhere to the below strict project-wide standards:

- **API-Centric Connectivity:** All functionality is exposed via standards like REST services (OpenAPI v3) and IoT protocols like MQTT or CoAP.
- **Identity and Access Management:** Centralized via Keycloak, supporting JWT tokens for humans and a tiered API Key system for machine-to-machine isolation.
- **Secure Asset Access:** Direct access to massive EO raster assets is managed via time-bound pre-signed URLs to ensure data sovereignty.

6.4 VALIDATION THROUGH PILOT SITES AND OPEN CALLS

The NOSTRADAMUS technical framework is validated through a rigorous practical implementation strategy that integrates diverse biogeographical, socio-economic, and technological contexts

across Europe. This process ensures that the architecture is not only theoretically sound but also resilient and effective in real-world agricultural environments.

The efficacy of this application model is validated through real-world scenarios. Validation scenarios are summarized below and further described next in this section:

- **Demonstration Sites:** Applications are deployed across five sites (Germany, Serbia, Switzerland, Slovenia, and Cyprus) to address local biogeographical needs.
- **Open Calls:** Funded third-party developers utilize the bootstrapping software to create interoperable applications, covering all aspects of the hybrid architecture.
- **Performance Evaluation:** Applications are evaluated based on their accuracy, reliability, sub-second latency, and energy performance in areas with weak connectivity.

6.4.1 REAL-WORLD VALIDATION VIA DEMONSTRATION SITES

The project utilizes five strategic demonstration sites, located in Germany, Serbia, Switzerland, Slovenia, and Cyprus, to serve as the primary living labs for testing core modules and applications. These sites were specifically selected to cover a wide spectrum of EU agricultural realities:

- Germany (DE): Represents the Continental region with a focus on high-productivity cereal land while validating performance in areas with low average internet speeds.
- Serbia (RS): Covers the Pannonian region, focusing on smallholder farmers and fragmented land management to ensure digital solutions are inclusive for non-industrial producers.
- Switzerland (CH): Provides an Alpine context with high-tech infrastructure, used for validating module replication and the integration of advanced machinery data.
- Slovenia (SL): Another Pannonian context, emphasizing the improvement of competitiveness for farmers with low bargaining power through data-based decision support.
- Cyprus (CY): Represents the Mediterranean region, which faces the project's most significant challenges: the lowest cereal yields, highest fertilizer consumption, and critical connectivity constraints.

These fields facilitate a Multi-Actor Approach (MAA), connecting IT and domain experts in a continuous co-creation process.

6.4.2 DEMOCRATIZING DEVELOPMENT THROUGH OPEN CALLS

Validation is extended to the broader European innovation ecosystem through two Open Calls (WP6), funding 4–8 projects led by SMEs and startups. Funded third parties utilize the platform's bootstrapping software (e.g. the IoT SDK) and the Low-Code Platform to rapidly synthesize applications.

- **Architectural Compliance:** Successful applicants are required to develop digital solutions that cover all three aspects of the hybrid architecture, ensuring the full System of Systems is utilized and tested.

- **Interoperability and Openness:** All applications must remain interoperable with the NOSTRADAMUS ecosystem, utilizing the established Data Infrastructure (I) and IoT Observatory (II) while committing to open-source principles.

6.4.3 PERFORMANCE EVALUATION

To ensure the high Veracity and impact of the platform, applications undergo a comprehensive evaluation based on three pillars:

1. **Technical Performance:** Measuring accuracy, reliability, usability, and throughput time, with a specific focus on achieving *sub-second latency* in real-time streams.
2. **Socio-economic Impact:** Conducting cost-benefit analyses and measuring effects on farmer income, livelihoods, and digital literacy to ensure the solutions are financially viable and easy to adopt.
3. **Environmental Impact:** Evaluating short-term effects on biodiversity and soil health while quantifying the i*mproved energy balance*. The project targets significant sustainability milestones, including a *>15% increase in crop yields*, a *>40% reduction in pesticide use*, and a *≤15% energy saving in production* through data-driven optimization.

The findings from these pilots are used to generate zoning maps. These maps identify other EU regions with similar bio-geographical and agroclimatic conditions where the validated digital applications can be replicated with similar positive outcomes, ensuring the long-term scalability of the NOSTRADAMUS approach.

7 INFRASTRUCTURE & OPERATIONS

The physical foundation of the NOSTRADAMUS platform utilizes a high-performance, on-premises cloud infrastructure designed to support compute-intensive EO processing and high velocity IoT streams. The deployment strategy leverages a modern virtualization layer via Proxmox technology and a lightweight Kubernetes (K3s) orchestration layer to provide a container-native environment. This infrastructure is specifically tuned to the project's storage requirements, implementing a tiered model that utilizes S3-compatible object storage for massive raster assets and high-performance block storage for distributed NoSQL databases.

7.1 PHYSICAL INFRASTRUCTURE & VIRTUALIZATION STRATEGY

The physical foundation of the NOSTRADAMUS platform utilizes a high-performance on-premises cloud infrastructure. It is designed to support compute-intensive EO processing and high-velocity IoT streams. The deployment strategy leverages a modern virtualization layer via Proxmox technology and a lightweight Kubernetes (K3s) orchestration layer to provide a container-native environment. This infrastructure is specifically tuned to the project's storage requirements. It implements a tiered model that utilizes S3-compatible object storage for massive raster assets and high-performance block storage for distributed databases.

Virtualization Layer: Proxmox VE acts as the hypervisor managing the underlying hardware. The total hardware pool includes 192 Cores, over 1.2TB of total RAM, and a raw storage capacity of approximately 1.1PB.

Orchestration Layer: K3s (Lightweight Kubernetes) is the chosen orchestrator running on top of Proxmox. This aligns with the project requirement for a cloud-native approach and a container orchestration environment, as identified in the Data Hub architecture.

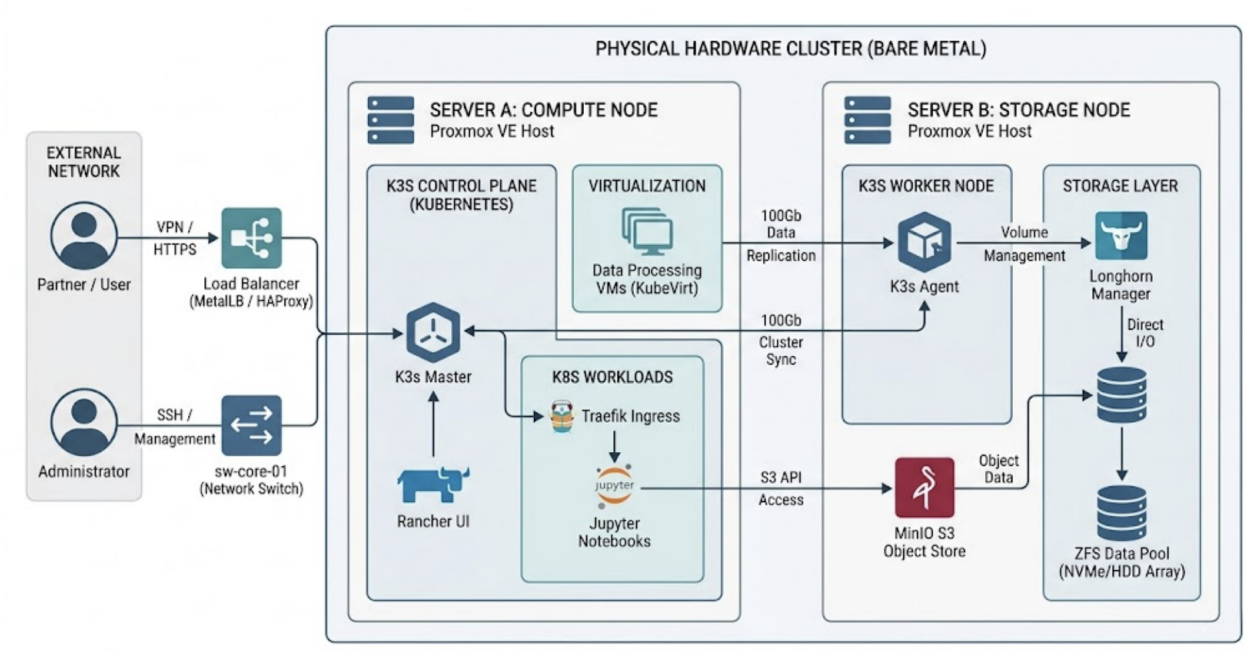

FIGURE 24: PHYSICAL HARDWARE CLUSTER & VIRTUALIZATION STRATEGY

7.1.1 SPECIFICATIONS

The Physical Hardware Cluster (PHC) of NOSTRADAMUS forms the core on-premises infrastructure. It is designed to manage and process demanding Earth Observation (EO) data and high-velocity Internet of Things (IoT) data streams. The setup allows for granular control over resources.

The physical hardware cluster is logically divided into specialized nodes to optimize for different workloads:

- **Compute Node (GPU Server)**: This server is dedicated to computational tasks and AI workloads. It features 2x AMD Genoa 9474F CPUs, 1.1TB of RAM, and 4x NVIDIA Ada L40S 48GB GDDR6 GPUs.

- **Storage Node**: This server focuses on data storage and management. It features 2x Intel Xeon Platinum 8352V CPUs and 128GB of RAM.
    - **Capacity**: Exactly 786TB of usable capacity.

- **Data Protection**: RAID 6 configuration with 4 hot spares across 48x Seagate 24TB HDDs.
- **Cache and OS**: 2x Samsung PMSA3 960GB NVMe drives configured in RAID 1.

On top of the bare-metal hardware, the virtualization layer is built using Proxmox. This allows for the on-demand deployment of public or private VM instances, to host algorithms, modules, tools and services to support the development of applications in the context of NOSTRADAMUS.
Furthermore, a Kubernetes environment is built on top of the Proxmox virtualization layer, to provide orchestration to DataHub, IoTO and other components of the NOSTRADAMUS overall architecture. This layer is built as a fully capable k3s stack. The table below provides the specifications of the bare-metal infrastructure.

TABLE 11: SPECIFICATIONS OF THE NOSTRADAMUS ON-REMISE PHC

Category	Parameter	Specification Value	Project Justification & Source
1. Storage	**RAID Configuration**	**RAID 6 (or ZFS RAIDZ2)**	**Data Protection:** Ensure fault tolerance against double drive failure.
	Target Usable Capacity	**786 TB 48x Seagate 24TB HDD**	**High Volume Storage:** Required for storing massive Earth Observation (EO) archives (Sentinel/Landsat) and Cloud Optimized GeoTIFFs (COGs) for the Data Cubes and Data Hub.
	Storage Type	**Object Storage (S3-compatible) & Block Storage**	**Architecture Alignment:** The Data Hub and Data Cubes require S3 for EO assets (COGs) and Block storage for high-performance databases (Cassandra/PostgreSQL).
	Cache	**2x Samsung PMSA3 960GB NVMe drives configured in RAID 1**	**Performance Acceleration & High Availability:** The dual NVMe SSD cache, configured in a RAID 1 mirror, serves as a high-speed buffer for both read and write I/O operations. This dramatically accelerates access to frequently used data from the main object and block storage, which is critical for the demanding, real-time workloads of the Data Hub's CWL workflows and the stream processing engine (Flink/Kafka). The RAID 1 configuration ensures redundancy, protecting against

			cache drive failure and ensuring uninterrupted service.
2. Compute (Kubernetes)	**Virtualization Layer**	**Proxmox**	**Management:** Provides the required virtualization layer to deploy and expose both VM instances and containers as requested.
	Allocation Strategy	**Primary Resource Pool (~85%)**	**Cloud-Native Design:** The architecture is fundamentally container-based (Docker), requiring orchestration for scalability,
	Reserved Cores	**160 Cores**	**Heavy Processing:** Supports concurrent execution of CWL workflows (Data Hub) and real-time stream processing (Flink/Kafka).
	Reserved RAM	**850 GB**	**In-Memory Processing:** Essential for Apache Flink state management and Open Data Cube indexing operations.
	Orchestrator	**K3s (Lightweight Kubernetes)**	**Efficiency:** Selected layer for container orchestration to manage microservices (API, GUI, Processing).
(VMs)	**Allocation Strategy**	**Secondary Resource Pool (~15%)**	**Legacy & Management:** Reserved for components not yet containerized or for specific edge-device simulations.
	Reserved Cores	**32 Cores**	**Management Plane:** Sufficient for hosting the Proxmox control plane, legacy databases etc.
	Reserved RAM	**150 GB**	**Stable Services:** Dedicated memory for persistent management services that do not require dynamic scaling.

Next, we provide a comprehensive description of the compute and storage nodes, while network connectivity and data flow are also covered.

7.1.2 COMPUTE NODE

The Compute Node includes the following layers and components:

- **Proxmox VE Host**: Acts as the hypervisor managing the underlying hardware.
- **Kubernetes (K3s) Control Plane**: A lightweight Kubernetes distribution used as the orchestrator.
- **Network Interface**: Flannel is used as the container network interface (CNI).

- **Ingress and High Availability**: The cluster uses an HAProxy frontend with a Virtual IP (VIP) to ensure high availability.
- **Load Balancer**: MetalLB is deployed to manage load balancing across the cluster.
- **Management UI**: KubeSphere provides a centralized user interface for managing Kubernetes clusters and monitoring operations.

7.1.3 STORAGE NODE

This server is primarily focused on data persistence and management, offering different storage types for diverse data needs. More specifically it hosts:

- **Proxmox VE Host:** Similar to the compute node, a Proxmox VE Host provides the virtualization layer on the storage node
- **K3s Worker Node:** This node functions as a worker within the Kubernetes cluster, hosting agents specifically for volume management
    - K3s Agent: Connects the worker node to the K3s Master, enabling it to receive and execute workloads, particularly storage-related tasks
    - Volume Management (Longhorn Manager): Longhorn provides distributed block storage specifically for Kubernetes. It manages persistent volumes, ensuring that applications have reliable and scalable storage accessible from anywhere within the cluster
- **Storage Layer:** This layer is optimized for different data types and access patterns.
    - Direct I/O: The inclusion of Direct I/O implies high-performance access to storage devices, minimizing overhead and maximizing data throughput for demanding applications
    - Object Data / MinIO S3 Object Store: MinIO, an S3-compatible object storage solution, is employed for storing massive raster assets. This is crucial for Earth Observation (EO) archives, which often consist of large, static files like Cloud Optimized GeoTIFFs (COGs). Object storage is ideal for its scalability, cost-effectiveness, and ability to handle unstructured data.
    - ZFS Data (NVMe/HDD Array): ZFS is utilized for high-performance block storage, configured over an array that combines fast NVMe (Non-Volatile Memory Express) drives with traditional Hard Disk Drives (HDDs). This tiered approach allows for balancing performance and capacity, making it particularly suitable for distributed NoSQL databases such as Apache Cassandra, which require high I/O throughput and reliable persistence for IoT event streams The storage system is configured with RAID 6 (or ZFS RAIDZ2) to meet data replication requirements and ensure fault tolerance, aiming for a usable capacity of ~800-900 TB.

7.1.4 NETWORK CONNECTIVITY AND DATA FLOW

The various components of the cluster are interconnected through high-speed networks to facilitate efficient data transfer. The network configuration enforces strict rules to maintain security and prevent conflicts:

- **Access Rules**: Management of access to the infrastructure requires a Cisco Secure Client VPN connection. However, public IPs are available to allow end-user access to the deployed services.
- **IP Allocation**: MetalLB uses the IP pool 172.16.63.180 to 199. For Virtual Machines, static IPs in the range of 172.16.63.40 to 172.16.63.150 must be used to avoid network conflicts.
- **OS Images**: The Proxmox environment supports a variety of OS templates, allowing flexible application deployments beyond standard Ubuntu images.
- **Network Speeds**: High-bandwidth 100Gbits connections are supported for data traffic between the compute and storage nodes.

7.1.5 OVERALL SYSTEM INTEGRATION AND TECHNOLOGIES

This infrastructure supports the NOSTRADAMUS Cloud Platform, which includes three main functional pillars:

- IoT Observatory (I): This acts as a Big Data Management System (BDMS) designed to ingest, buffer, and process high-velocity sensor data streams with sub-second latency. It leverages Apache Kafka for persistent buffering of data streams and Apache Cassandra for long-term storage of IoT event streams.
- Data Hub (II): This functions as the system's "Batch Layer" and long-term storage engine, managing massive static datasets, EO archives, and socio-economic indicators. It utilizes a STAC to index EO assets and derived products and stores raster data in COG formats to allow efficient retrieval of specific file segments. Data ingestion and processing workflows are defined using CWL and executed within Docker containers orchestrated by Kubernetes.
- Low-Code Platform (III): This layer facilitates the development of open-source digital applications by domain experts without requiring deep programming knowledge. can help domain experts create models through natural language descriptions.

The deployment strategy leverages Proxmox for virtualization and K3s (Lightweight Kubernetes) for orchestration, providing a container-native environment. This robust setup ensures high availability, scalability, and efficient data management for the NOSTRADAMUS project's agricultural applications, providing a container-native environment. This robust setup ensures high availability, scalability, and efficient data management for the NOSTRADAMUS project's agricultural applications.

8 CONCLUSIONS

The present deliverable, Nostradamus Deliverable number D2.3, entitled “D2.2 Software Architecture for Cloud, Edge and Mixed Solutions-M20”, establishes the comprehensive technical blueprint for the Nostradamus digital ecosystem. It successfully defines a modular and scalable Hybrid Edge-to-Cloud architecture designed to bridge the gap between physical agricultural assets and digital decision support. By decomposing every application into three interoperable functional tiers, App-Edge (A), App-Logic (B), and App-GUI (C), the architecture ensures a strict separation of

concerns while maintaining high performance across the entire resource continuum. This framework is specifically engineered to manage the "four V's" of agricultural big data Volume, Variety, Velocity, and Veracity, reconciling the need for sub-second reactive intelligence in the field with high-precision analytical intelligence in the cloud.

A core achievement of this deliverable is the specification of the Low-Code Application Platform (III). Utilizing Model-Driven Engineering and specialized Domain-Specific Languages, the platform democratizes the creation of digital tools, allowing domain experts to participate in software development through high-level abstractions. The innovative integration of AI-assisted development and the Vibe Coding paradigm further lowers the technical entry barrier, enabling the rapid and secure synthesis of applications tailored to local biogeographical conditions, even in environments with weak connectivity.

Furthermore, the architecture provides a robust operational foundation for the project's four pillars: the Data Hub for EO data lifecycle management, the IoT Observatory for real-time event streaming via Apache Kafka, the Data Cubes for multi-dimensional spatiotemporal analysis, and the Low-Code Platform for automated software generation. These pillars are supported by rigorous infrastructure standards, including 99% system availability and sub-second latency for time-critical agricultural alerts.

The finalization of D2.3 marks a critical transition from the initial requirements elicitation phase to the detailed design and prototyping of the system's core components. The roadmap for the subsequent development phases of the project focuses on the following key actions:

- Finalization of the AGE Prototype: Work in Task 2.4 will focus on the implementation of the Application Generation Engine (D2.5) to provide the technical foundation for third-party developers.
- System Integration and Calibration: Coordinating with WP3, WP4, and WP5 to consolidate data pipelines, calibrate the first core agricultural modules, and instantiate the five national Data Cubes for the demonstration sites.
- Operational Validation via Open Calls: Utilizing the established architecture and bootstrapping software to support the WP6 Open Calls, where funded SMEs and startups will build and test interoperable applications across the edge-to-cloud continuum.
- Knowledge Transfer and Capacity Building: Developing comprehensive training materials and recorded demonstrations to empower IT experts and domain stakeholders to effectively leverage the platform's analytical capabilities.

By adhering to open-source principles and standard interfaces (OpenAPI, OGC, STAC, CWL, ODC etc.), the NOSTRADAMUS architecture not only addresses the project's immediate objectives for Food Security and European Independence but also establishes a resilient and inclusive pattern for the future of digital agriculture in the EU.