Optimisation of digital catch monitoring and reporting in European Fisheries

D4.1: Requirements Specification and System Architecture

Responsible Author:
Antonis Koukourikos (SCiO)

Co-funded by
the European Union

optifish.eu

Document Information

Grant Agreement No.	**101136674**		
Project Acronym	**OptiFish**		
Project Title	Optimisation of digital catch monitoring and reporting in European Fisheries		
Type of Action	HORIZON Innovation Actions		
Call	HORIZON-CL6-2023-FARM2FORK-01		
Start – Ending date	1 February 2024 – 31 January 2028	**Duration**	48 months
Project Website	optifish.eu		
Work Package	WP4: Data management framework and system architectures		
WP Lead Beneficiary	SCiO P.C. (SCiO)		
Relevant Task(s)	T4.1: Requirements specification: Data management framework and system architectures		
Deliverable type [1]	R	**Dissemination level** [2]	PU
Due Date of Deliverable	31 July 2025		
Submission Date	31 July 2025		
Responsible Author	Antonis Koukourikos (SCiO)		
Contributors	Josean Fernandes (AZTI), Lancelot Blondeel (EV ILVO), Teun De Boer (EFICE)		
Reviewer(s)	Jade Maes (EV ILVO)		

[1] *Please consult the Grant Agreement: R: Document, report; DEM: Demonstrator, pilot, prototype, plan designs; DEC: Websites, patents filing, press & media actions, videos, etc.; DATA: Data sets, microdata, etc; DMP: Data management plan; ETHICS: Deliverables related to ethics issues; SECURITY: Deliverables related to security issues; OTHER: Software, technical diagram, algorithms, models, etc.*

[2] *Please consult the Grant Agreement: PU – Public, fully open, e.g. web (Deliverables flagged as public will be automatically published in CORDIS project's page); SEN – Sensitive, limited under the conditions of the Grant Agreement; Classified R-UE/EU-R – EU RESTRICTED under the Commission Decision No2015/444; Classified C-UE/EU-C - EU CONFIDENTIAL under the Commission Decision No2015/444; Classified S-UE/EU-S – EU SECRET under the Commission Decision No2015/444.*

Document History

Version	Changes	Date	Contributor
0.1	ToC	20/06/2025	Antonis Koukourikos (SCiO)
0.3	Introduction & Section 2	27/07/2025	Antonis Koukourikos (SCiO)
0.6	Section 3	14/07/2025	Josean Fernandes (AZTI) Lancelot Blondeel (EV ILVO) Teun De Boer (EFICE)
0.8	Full draft finalised and submitted for internal review	21/07/2025	Antonis Koukourikos (SCiO)
0.9	Internal Review Completed	31/07/2025	Jade Maes (EV ILVO)
1.0	Final version ready for submission	31/07/2025	Antonis Koukourikos (SCiO)

OptiFish Consortium

No.	Participant Organisation Name	Short Name	Country
1	EIGEN VERMOGEN VAN HET INSTITUUT VOOR LANDBOUW- EN VISSERIJONDERZOEK	EV ILVO	BE
2	FUNDACION AZTI - AZTI FUNDAZIOA	AZTI	ES
3	BENCO BALTIC DOO ZA SAVJETOVANJE IUSLUGE	BENCO	HR
4	DANMARKS TEKNISKE UNIVERSITET	DTU	DK
5	REFRAME FOOD ASTIKI MI KERDOSKOPIKI ETAIRIA	RFF	EL
6	SCIO IKE	SCiO	EL
7	STICHTING WAGENINGEN RESEARCH	WR	NL
8	UNIVERSITY OF CUKUROVA	UC	TR
9	FISKERIDIREKTORATET	NDF	NO
10	SINTEF OCEAN AS	SO	NO
11	ELECTRONIC FISH INFORMATION CENTRE EUROPE B.V	EFICE	NL
12	Justervesenet	JV	NO
13	VCU ROBOTICS B.V.	VCUR	NL
13.1	VCU TCD B.V.	VCU	NL
14	WAGENINGEN UNIVERSITY	WU	NL
15	ANCHOR LAB KS	ANCHOR	DK
16	DANMARKS PELAGISKE PRODUCENTORGANISATION FORENING	DPPO	DK
17	ZUNIBAL SL	ZUN	ES
18	DANMARKS FISKERIFORENING PRODUCENTORGANISATION	DFPO	DK

Executive Summary

This deliverable presents the architectural design of the OptiFish Data Platform, developed within Work Package 4 (WP4) of the OptiFish project. The platform provides the technical backbone for the secure, scalable, and interoperable management of data related to fisheries monitoring, control, and decision support. Its purpose is to enable the integration, processing, semantic harmonisation, and delivery of data collected from a wide range of sources, including onboard electronic monitoring systems, sensor feeds, vessel tracking systems (e.g. AIS, VMS), and third-party providers.

The architecture has been designed following the C4 model, describing the platform at the context, container, and component levels. This approach ensures clarity, modularity, and alignment with both technical requirements and stakeholder expectations. The design was informed by a structured requirements elicitation process, which combined internal partner consultations, stakeholder engagement, and questionnaire-based data collection. This process captured the diverse functional, compliance, and user interface needs of scientists, regulators, and fishers.

The main characteristics of the resulting architecture include:

- A modular ETL pipeline for data ingestion, validation, and semantic enrichment;
- Data storage and persistence mechanisms to support both raw data access and curated analysis;
- A suite of analytical and geospatial services
- Secure, role-based dashboards and APIs tailored to distinct stakeholder groups;
- A robust audit and access control layer ensuring GDPR compliance and traceability.

In addition, the platform is built to serve as an integration point for analytics, AI-based tools, and decision systems developed in other work packages. In summary, the report defines a flexible and future-proof platform capable of evolving with project needs while remaining grounded in current best practices for data governance, interoperability, and open science.

Contents

List of Figures

List of Tables

List of Abbreviations

MCS	Monitoring, Control and Surveillance	**EM**	Electronic Monitoring
C4	Context, Container, Component, Code	**EFCA**	European Fisheries Control Agency
VMS	Video Monitoring System	**AIS**	Automatic Identification System
REM	Remote Electronic Monitoring	**FLUX**	Fisheries Language for Universal Exchange
API	Application Programming Interface	**ETL**	Extract-Transform-Load
ML	Machine Learning		

1 Introduction

The OptiFish project aims to enhance the Monitoring, Control, and Surveillance (MCS) of fisheries through the use of advanced digital technologies, including Electronic Monitoring (EM), machine learning, geospatial analytics, and integrated decision support systems. These tools will help ensure sustainable fishing practices, compliance with EU regulations, and support for various stakeholders, including fishers, scientists, and regulatory authorities. Work Package 4 (WP4) plays a central role in realizing this vision by designing and implementing the OptiFish Data Platform — the technical backbone that enables the ingestion, processing, storage, and exploitation of fisheries data from heterogeneous sources.

To achieve this, WP4 aims to address several technical challenges:

- Aggregation of data from multiple onboard and remote sources, including video feeds, sensors, logbooks, and satellite data.
- Implementation of a modular and extensible architecture that supports the evolving needs of machine learning, semantic enrichment, and real-time inference and visualisation.
- Ensuring data privacy, access control, and legal compliance, particularly in accordance with the GDPR and domain-specific policies.
- Serving as a common infrastructure to support work on other WPs through interoperable APIs and shared data services.

The present report describes the architecture of the computational infrastructure that will serve the data needs of OptiFish. The architectural design process followed a concrete, established methodology, while taking into account industry best practices and background knowledge, while keeping a certain degree of flexibility and adaptability to changes. It also incorporates design decisions based on functional and non-functional requirements, technical dependencies on other work packages, and the general principles of modularity, traceability and scalability to support long-term usage and integration. The architecture is intended as a reference for both internal development and external engagement and lays the foundation for the subsequent implementation, deployment and incremental refinement of the platform.

This deliverable is organised as follows. Section 2 presents the approach used to collect stakeholder input and elicit the initial set of requirements for the platform, along with the main outcomes of the requirements elicitation process; Section 3 describes the adopted methodology for establishing the architectural design of the data platform, and the resulting architecture from applying the methodology against the collected requirements; Section 4 concludes the deliverable and indicates the next steps for both the operationalisation of the architectural design and the process for refining and updating it as requirements evolve and become more clearly defined.

2 Requirements Specification and Analysis

The requirements elicitation followed a multi-pronged approach, involving both internal technical coordination and external stakeholder consultation:

Structured Questionnaires: A suite of targeted questionnaires was developed and distributed internally to consortium partners and technical leads. These were tailored to elicit:

- Expected data sources, formats, and volumes
- Functional needs for data ingestion, transformation, and visualisation
- Access control and compliance requirements
- Integration expectations for tools under development

Direct Partner Engagement: One-on-one and group discussions were held with partners responsible for key OptiFish components, such as:

- Video and ML-based monitoring tools
- Analytics and risk models
- Decision support and user-facing dashboards
- Use-case design and pilot validation

Stakeholder Representative Input: WP4 also consulted representatives of end-user communities (e.g. regulatory authorities, fishers, and scientific advisors) to capture domain-specific expectations and constraints. These consultations helped shape the role-based access design, prioritisation of reporting functionalities, and visualisation customisation options.

The requirements collection process led to the identification of the core operations to be supported by the data platform:

- **Data Ingestion**: Gather data from various sources in real-time or batch mode
- **Data Storage**: Store raw and processed data for analysis, reporting and provenance
- **Data Processing and Transformation**: Prepare data for analytics, perform transformations, and train/apply AI models
- **Data Governance**: Ensure data security, privacy and compliance with relevant regulations
- **Analytics and Visualisation**: Provide actionable insights through dashboards, reports, and visual analytics
- **Integration and Programmatic Access**: Enable external applications to access data and analytical results
- **User Access and Interfaces**: Provide a user-friendly interface for different stakeholders to interact with the platform, upload data, access analytical results, generate reports.

Additionally, requirements defined a number of cross-cutting needs and priorities that directly informed and shaped the architectural structure:

- The need for a multi-source ingestion layer capable of handling structured data (e.g. logbooks), unstructured data (e.g. video), and sensor feeds in both real-time and batch modes.

- The importance of semantic enrichment and standardisation, due to the heterogeneity of data sources and regulatory jurisdictions.
- The requirement for role-specific interfaces and access control mechanisms, with strong emphasis on GDPR compliance and auditability.
- The need to support interoperable APIs to integrate analytics services and domain tools, while exposing usable dashboards for decision support.
- The necessity of providing both raw data access (for scientific stakeholders) and curated reporting views (for compliance officers and fishers).

Finally, non-functional needs mainly around scalability and flexibility informed architectural and technical choices for the implementation and deployment of the components entailed in the platform. The detailed design responding to the platform's scope and requirements is presented in the following section.

3 Architectural Design

The architectural description of the OptiFish Data Platform adopts the C4 model[3], a widely used visual framework for representing software architecture across multiple levels of abstraction. Originally developed by Simon Brown, the C4 model allows architects to communicate the structure and design of complex systems to both technical and non-technical stakeholders, with a strong emphasis on clarity, modularity, and traceability.

The four levels of the C4 model include:

Context (Level 1): Describes the system as a whole and how it interacts with users, external systems, and other stakeholders.

Container (Level 2): Breaks down the system into major technology units or "containers" such as applications, services, databases, or APIs — each with a defined responsibility and communication flow.

Component (Level 3): Describes the internal structure of each container, detailing the main components (e.g. services, modules, libraries) and their responsibilities.

Code (Level 4): Describes the internal implementation of a specific component at the class or code level. This is out of scope for the current deliverable.

By applying this model, the architecture of the OptiFish Data Platform is made transparent and accessible while remaining technically rigorous. It supports alignment with project requirements and enables focused collaboration across development, integration, and stakeholder-facing work packages.

The subsections that follow provide a detailed breakdown of the OptiFish Data Platform at the context, container, and component levels as defined and implemented during the first 18 months of the project.

[3] https://c4model.com

3.1 Context Level

At the context level, the OptiFish Data Platform is positioned as the central infrastructure enabling the collection, processing, storage, and dissemination of data across the OptiFish ecosystem. More specifically, the OptiFish Data Platform is a cloud-based fisheries data management system that integrates multiple data sources, processes information using AI models, and provides analytics for industry stakeholders and regulators.

Key Actors and Interactions

- Fishers: Use the system for real-time fishing data, quota tracking, and operational insights.
- Regulatory Authorities (e.g., EFCA, national agencies): Access compliance data, reports, and alerts to monitor fisheries activities.
- Marine Scientists: Utilize historical and real-time data for research on fisheries sustainability.

Key external communications

- Vessel Monitoring & Sensor Systems: Provide VMS, AIS, REM, catch, fuel consumption, vessel state and environmental data for analysis.
- External Systems & Data Providers: Integrate satellite imagery, FLUX reports, and other industry datasets.
- Copernicus: Supply environmental data forecast and fill missing historical data gaps.
- EMODnet data

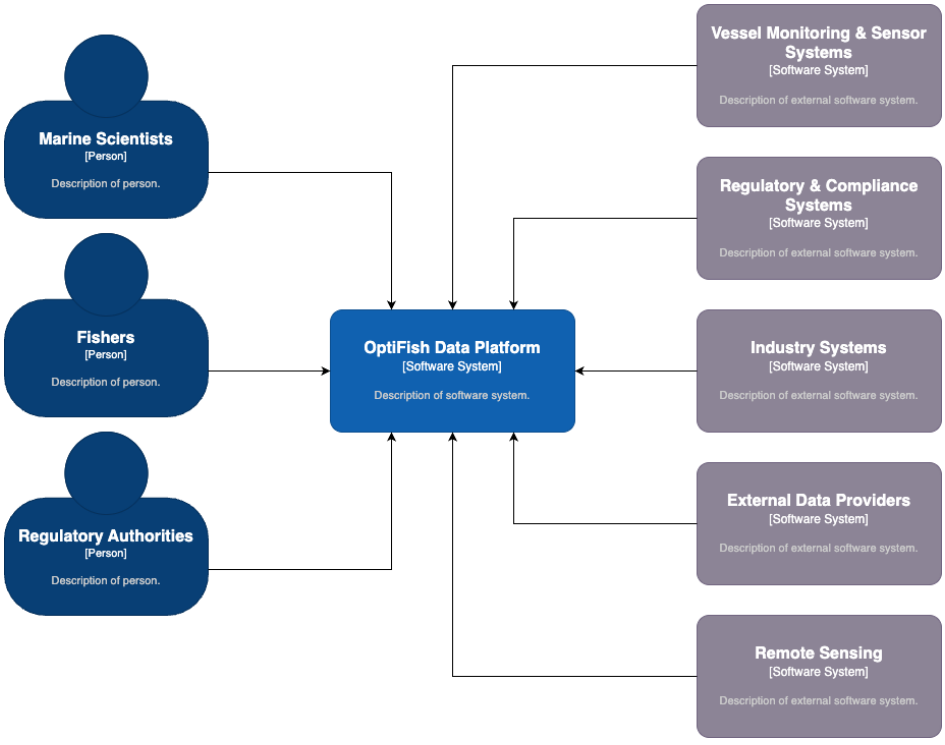

Figure 1: OptiFish Architecture - Context Level

The system will operate as a centralized data hub, ingesting, processing, analysing, and visualising data for all stakeholder types and end-user personas.

3.2 Container Level

The platform is divided into multiple logical containers, each representing a distinct application or data service with its own responsibility, deployment model, and technology stack. The core containers foreseen in the platform are summarised in the following table.

Table 1: OptiFish Data Platform Architecture C2 Containers

Container	Description	Related Operation(s)
Web Portals	Provides user dashboards for different stakeholder types	User Interface and Access
API Gateway	Central entry point for external system, sensor data management, and API requests	Data Ingestion, Integration and Programmatic Access
Data Lake	Storage of unstructured data	Data Storage
Data Warehouse	Storage of structured data	Data Storage
ETL Pipeline	Extract/Clean/Transform raw data	Data Processing and Transformation
Analytics Engine	Bootstraps and executes analytical and AI models	Data Processing and Transformation, Analytics and Visualisation
Geospatial Engine	Handles the analysis of geospatial data	Data Processing and Transformation, Analytics and Visualisation
Access Control and Identity Management	Ensures data security, role-based access, and regulatory compliance	Governance
Audit and Logging Service	Tracks data access, modification, and compliance events	Governance
Data Export Service	Allows users to export reports and insights	User Interface and Access, Integration and Programmatic Access

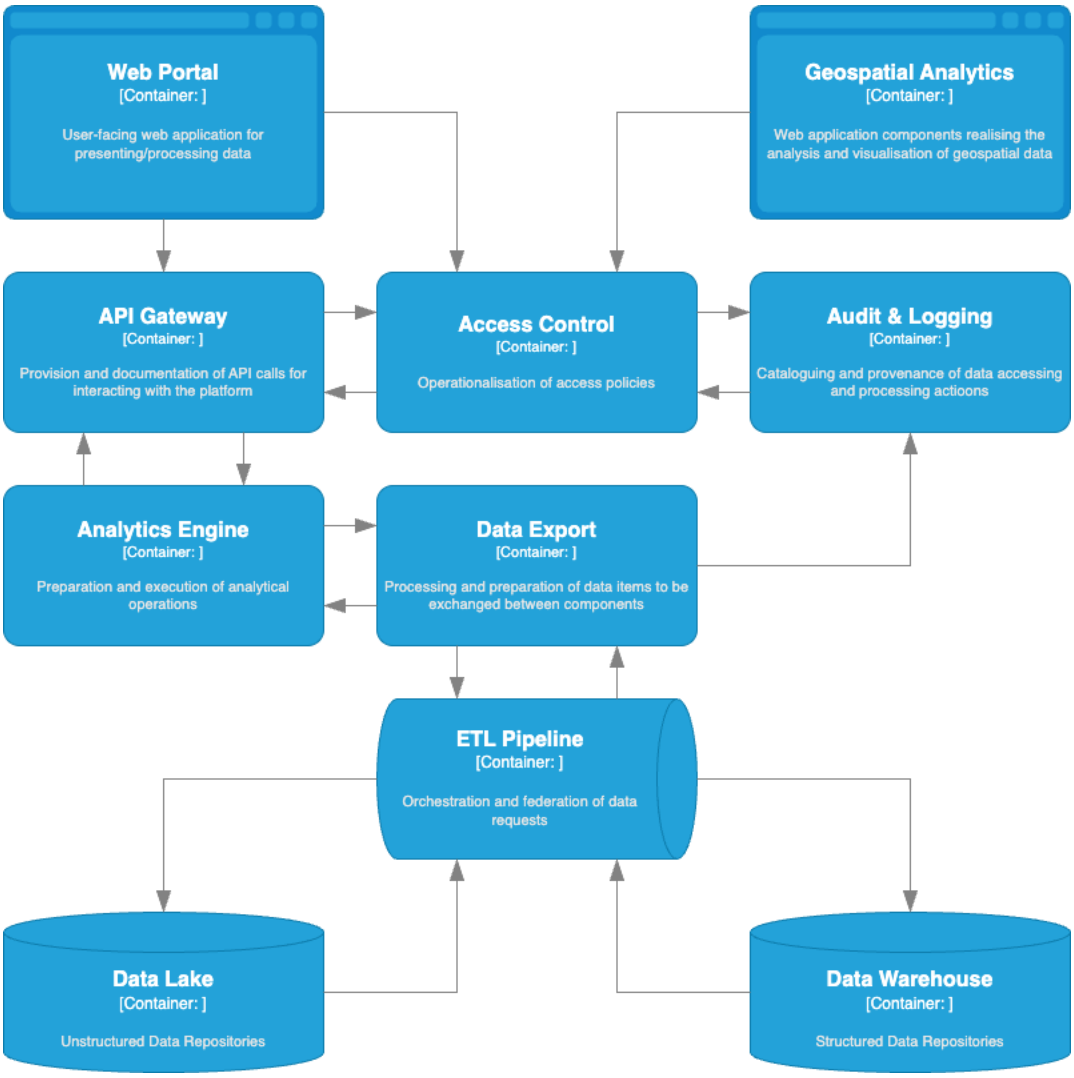

Figure 2: OptiFish Architecture - Container Level

3.3 Component Level

Each container is internally decomposed into core components, each with a single responsibility and well-defined interfaces. Below is a representative sample from selected containers.

3.3.1 ETL Pipeline

Table 2: Data Platform Components: ETL Pipeline Container

Component	Responsibilities	Technologies
Ingestion Service	Connects to data sources (vessels, EM systems, external APIs)	Apache NiFi, Kafka Connect

Validation Module	Schema and integrity checks on incoming data	JSON Schema, Apache Beam
Data Cleaner	Applies cleaning rules (removal, fixing, standardisation)	Python (Pandas), Spark
Format Converter	Transforms data into standard formats	Apache Camel, custom scripts
Semantic Enricher	Adds metadata, tags, ontology references	Python, RDFLib, SPARQL
Pipeline Orchestrator	Coordinates ETL steps, logs execution	Apache Airflow, Prefect

3.3.2 API Gateway

Table 3: Data Platform Components: API Gateway Container

Component	Responsibilities	Technologies
Routing Engine	Routes requests to backend services	Spring Cloud Gateway, Kong
Rate Limiter	Controls usage per user/API key	NGINX, Envoy, Redis
Auth Adapter	Validates tokens and forwards user claims	Keycloak Adapter, OAuth2 Proxy
Request Validator	Validates payloads against API specs	OpenAPI Validator, JSON Schema
Monitoring Hook	Captures API metrics and errors	Prometheus, Grafana, ELK

3.3.3 Access Control

Table 4: Data Platform Components: Access Control Container

Component	Responsibilities	Technologies
User Directory	Stores users, roles, credentials	Keycloak, LDAP
RBAC Service	Enforces fine-grained permissions	Keycloak Authorization, OPA
Token Service	Issues and verifies JWT tokens	OAuth2, OpenID Connect
Compliance Checker	Enforces data governance constraints	Custom policy engine, OPA
Audit Hook	Logs auth events to audit container	Kafka, Fluentd

3.3.4 Analytics Engine

Table 5: Data Platform Components: Analytics Engine Container

Component	Responsibilities	Technologies

Model Executor	Runs ML models on incoming or stored data	Python, TensorFlow, PyTorch
Job Scheduler	Schedules and triggers analytics runs	Apache Airflow, Kubernetes CronJobs
Data Preprocessor	Prepares inputs for model runs	Pandas, Spark
Model Registry	Stores and version-controls ML models	MLflow, DVC
Result Publisher	Sends outputs to Data Export or Warehouse	Kafka, REST API

3.3.5 Geospatial Engine

Table 6: Data Platform Components: Geospatial Engine Container

Component	Responsibilities	Technologies
Map Renderer	Displays spatial layers, routes, events	Leaflet.js, Mapbox, OpenLayers
Spatial Processor	Computes zones, heatmaps, overlaps	PostGIS, GeoPandas
GeoAPI	Exposes geospatial queries	Flask, FastAPI, Node.js
Layer Manager	Handles base maps and user-defined overlays	GeoServer, QGIS Server

3.3.6 Web Portals

Table 7: Data Platform Components: Web Portals Container

Component	Responsibilities	Technologies
User Dashboard	Displays charts, alerts, KPIs	React, Vue, Angular
Data Viewer	Allows exploration of datasets	React Table, D3.js
Feedback Form	Sends feedback or corrections	Formspree, Custom Backend
Alert Panel	Displays compliance or safety alerts	WebSockets, MQTT
Profile Manager	Manages user settings and preferences	React + Auth integration

3.3.7 Data Export

Table 8: Data Platform Components: Data Export Container

Component	Responsibilities	Technologies
Export Generator	Builds reports in CSV, JSON, PDF	Python, Pandas, JasperReports

Report Scheduler	Automates periodic exports	Quartz, cron, Airflow
Data Formatter	Applies export templates and formatting	Jinja2, XSLT
Download API	Endpoint for users to retrieve files	Flask, SpringBoot

3.3.8 Audit & Logging

Table 9: Data Platform Components: Audit and Logging Service Container

Component	Responsibilities	Technologies
Log Collector	Captures logs from all services	Fluentd, Filebeat
Event Store	Stores structured events for auditing	Elasticsearch, Loki
Alert Generator	Flags suspicious or policy-violating actions	Kibana, Grafana Alerts
Log Dashboard	Interface for browsing logs	Kibana, Grafana, Graylog

3.3.9 Data Lake

Table 10: Data Platform Components: Data Lake Container

Component	Responsibilities	Technologies
Blob Storage	Holds large binary files	S3, MinIO, HDFS
Metadata Index	Indexes content types, timestamps, sources	ElasticSearch, PostgreSQL
Access API	Serves files or streams on demand	FastAPI, MinIO Gateway

3.3.10 Data Warehouse

Table 11: Data Platform Components: Data Warehouse Container

Component	Responsibilities	Technologies
Schema Manager	Defines and evolves warehouse schema	Liquibase, Flyway
Query Engine	Supports reporting and analytics	PostgreSQL, ClickHouse
ETL Loader	Loads curated data from ETL pipeline	Apache Sqoop, Spark

4 Conclusions and Future Work

The architectural work carried out in Work Package 4 during the first 18 months of the OptiFish project has laid the groundwork for a robust, modular, and interoperable data platform. By following the C4 model, we systematically defined the system's context, decomposed it into functional containers, and described the internal architecture of core components. This approach has ensured a high degree of clarity, traceability, and alignment with both user requirements and the technical needs of other work packages. It also helps to clarify component communication points, identify dependencies from external

systems and technologies, and select the appropriate candidate frameworks and technologies for development and deployment.

To ensure the successful evolution and deployment of the platform, the following design considerations will be systematically assessed as the platform transitions to its development phase:

- Iterative refinement of the architecture based on implementation experience, stakeholder feedback, and tool integration results.
- Formalisation of semantic services, including ontology and schema management, term mapping, and annotation APIs, to support interoperability and enrich data discovery.
- Development of compliance rule engines that encapsulate regulatory logic and support automated alerting and reporting.
- Scalable deployment planning, particularly for handling video-intensive use cases and multi-vessel data streams in real-world pilots.
- Security and privacy validation, with a focus on ensuring GDPR-compliant user access, data minimisation, and traceability across all platform layers.

Furthermore, the OptiFish data platform will take advantage of already established knowledge and assets for developing the components foreseen in the architecture: Firstly, we will examine bidirectional linkages with the EDITO European infrastructure[4] and particularly its DataLake services and APIs, for ingesting, processing and submitting data - under the relevant access and data exposure limitations will be established. Additionally, the semantic framework to be used for conceptualising and associating formally defined semantics for the data handled by OptiFish will be based on the PoseiDAT[5] open data schemas, extended accordingly to cover the needs of OptiFish.

In summary, the architecture defined in this deliverable provides a flexible and future-proof foundation for the OptiFish Data Platform. It reflects the project's commitment to technical excellence, user-centric design, and sustainable digital transformation in fisheries management.

[4] https://www.edito.eu
[5] https://poseidat.org/#/

END OF DOCUMENT