Management Solution for Experimental Data in Photosynthetic Research

University of Turku
Department of Computing
Master of Science (Tech) Thesis
Software Engineering
May 2026
Ujala Adil Yousuf

UNIVERSITY OF TURKU
Department of Computing

Ujala Adil Yousuf: Management Solution for Experimental Data in Photosynthetic Research

Master of Science (Tech) Thesis, 109 p.
Software Engineering
May 2026

Experimental research in photosynthetic sciences generates large volumes of heterogeneous data from instruments such as Micro Gas Chromatographs (Micro-GC), Membrane Inlet Mass Spectrometry (MIMS) systems, and Photobioreactors (PBR). These data are currently stored across different locations and formats, making systematic reuse, analysis, and traceability difficult. This thesis designs and justifies a structured, scalable data management system tailored for the Photosynthetic Microbes research group at the University of Turku.

The study adopts a Design Science Research (DSR) methodology. A structured literature review of existing research data management systems, including OMERO, SEEK, Dataverse, and OpenBIS, reveals persistent limitations in automation, interoperability, and usability for instrument-driven research settings. Building on these findings, a domain-specific Data Management System (DMS) is designed around a layered architecture: an automated Python-based ingestion layer that collects and normalizes instrument outputs; a relational storage layer hosted on institutional cloud infrastructure; and a web-based visualization and access layer for data browsing, upload, and visualization.

The system is evaluated against six functional and six non-functional requirements derived from the operational context of the research group. Key design decisions, including a format-agnostic measurement schema, flexible contextual data storage, automated provenance tracking, and hierarchical data organization, are grounded in established data management principles and FAIR guidelines. The evaluation confirms that the proposed system centralizes heterogeneous experimental data, reduces manual effort, and supports long-term data accessibility and reproducibility. The design principles underlying the system are transferable to other instrument-driven research settings facing comparable data management challenges.

Keywords: research data management, experimental data management, instrument-driven research, data automation, cloud storage, FAIR principles, design science research, data ingestion, visualization, open science

Acknowledgments

Completing this thesis has been a meaningful journey, and I am grateful to everyone who supported and guided me along the way.

First and foremost, I would like to thank my supervisor, Assistant Professor Tuomas Mäkilä, for his guidance and support throughout this research. His technical expertise, constructive feedback, and consistent encouragement to think critically about the design and evaluation of the system were invaluable at every stage of this work. I am particularly grateful for his patience in engaging with the many iterations of this thesis and for always pointing me toward a more rigorous and well-reasoned approach.

I would also like to extend my sincere gratitude to Dr. Sergey Kosourov, Senior Research Fellow in Molecular Plant Biology at the University of Turku, for serving as the domain expert for this research. His deep knowledge of photosynthetic research and the practical data challenges facing the Photosynthetic Microbes group gave the work its direction and purpose. His willingness to share insights from a biological sciences perspective, review the thesis content, and provide access to instrument data and workflows made it possible to design a system grounded in real research needs.

I am also grateful to the other members of the Photosynthetic Microbes research group, who were generous with their time and knowledge throughout this project. Their openness in describing their day-to-day data handling practices and experimental workflows was essential to understanding the problem the system was designed

to solve.

Finally, I acknowledge the open-source communities behind the tools and frameworks used in this work, whose contributions provided the technological foundation for the system developed in this thesis.

Turku, May 2026

Ujala Adil Yousuf

Contents

List of Figures

List of Tables

Glossary

Term	Definition
Cloud-Based DBaaS	A managed database hosting model in which infrastructure provisioning, maintenance, and backup are handled by a cloud provider. In this thesis, CSC Pukki provides PostgreSQL as a DBaaS within Finland's national research computing infrastructure.
Controlled Vocabulary	A standardized set of terms used consistently to label data attributes across different sources. Each ingestion script enforces a controlled vocabulary through an internal mapping dictionary that translates instrument-native variable names and units into standardized labels before database insertion.

Data Management System (DMS)	A software system designed to collect, organize, store, and provide access to data in a structured and consistent manner. In this thesis, the term refers specifically to the custom-built system developed for the Photosynthetic Microbes research group, which automates the ingestion, storage, and visualization of instrument-generated experimental data.
Design Science Research (DSR)	A research methodology in information systems that produces and evaluates purposeful artifacts, systems, models, or frameworks, as answers to identified practical problems, emphasizing iterative construction and grounded justification of design decisions.
Entity-Attribute-Value (EAV)	A data modeling approach in which each measurement is stored as a discrete row containing an entity identifier, an attribute name, and a corresponding value, rather than as a column in a wide table. This pattern is used in all seven measurement tables of the proposed system, enabling a consistent schema across instruments with different numbers and types of measured variables.

ETL (Extract, Transform, Load)	A data integration pattern in which raw data is extracted from one or more source systems, transformed into a consistent and structured format, and loaded into a target storage system. In this thesis, the ingestion pipeline follows the ETL pattern: instrument files are extracted from monitored directories, transformed into normalized Entity-Attribute-Value rows, and loaded into the PostgreSQL database.
FAIR Data Principles	A set of four guiding principles, Findable, Accessible, Interoperable, and Reusable, introduced by Wilkinson et al. to promote the quality and sustainability of research data.
File System Watcher	A software component that monitors directories on a file system and triggers a processing action when new files are detected. The `watchdog` Python library is used to implement event-driven ingestion in the proposed system.
Ingestion Pipeline	The automated sequence of processing steps that detects new instrument data files, validates their format, parses and normalizes their contents, and inserts the resulting records into the database.

JSONB	A binary-encoded JSON storage format supported by PostgreSQL, enabling efficient indexing and querying of semi-structured data. Used in the proposed system for experiment-level contextual attributes and the `raw_data` provenance column.
MD5 Hash	A fixed-length fingerprint computed from the contents of a file, used to verify file integrity and detect duplicate submissions during ingestion.
Metadata	Data that describes the context, content, and structure of other data. In this thesis, metadata is captured at the organizational, experiment, and measurement levels through relational columns and JSONB fields.
Provenance	The documented record of a dataset's origin, processing history, and transformations. Preserved in this system through the `raw_data` JSONB column and the `processed_files` audit log.
Research Data Management (RDM)	The active organization and maintenance of data produced or collected during a research project, spanning the full lifecycle from initial collection through long-term preservation. RDM encompasses technical infrastructure, metadata practices, governance policies, and access control mechanisms that together ensure data remains usable, traceable, and reproducible over time.

REST API	Representational State Transfer Application Programming Interface. An architectural style for networked interfaces in which clients interact with a server through standardized HTTP methods. In this thesis, a FastAPI-based REST API connects the React frontend to the PostgreSQL database.
Single-Page Application (SPA)	A web application architecture in which the interface is loaded once in the browser and subsequent interactions update the page dynamically without full reloads. The frontend in this thesis is implemented as a React SPA built with Vite.

1 Introduction

This thesis contributes a structured, scalable data management system customized for instrument-driven experimental research environments, integrating cloud storage and visualization tools within the University of Turku's infrastructure.

1.1 Background and Motivation

Experimental research in modern times produces large quantities of heterogeneous data. In the case of photosynthetic research, several instruments generate continuous datasets in different formats, including sensors for dissolved gases (Clark-type O_2 and H_2 electrodes), mass flow controllers and gas analyzers (MFC, GC, Micro-GC), mass-spectrometry systems for measuring gas exchange (Membrane Inlet Mass Spectrometry), and microprocessor-controlled photobioreactors (PBRs) used to monitor culture growth. The data produced from these instruments are essential for researchers to derive scientific conclusions from. Proper management of these heterogeneous data is critical: without it, systematic reuse, reproducible analysis, and long-term traceability of experimental results are difficult to achieve. Unfortunately, these data are currently stored in different formats across different locations, making centralized access, structured analysis, and consistent record-keeping troublesome and time-consuming.

1.2 Problem Statement

In many experimental research environments, data generated during experiments continue to be managed through fragmented and informal practices, including decentralized storage, personal file hierarchies, and ad hoc sharing mechanisms. While such approaches may support short-term needs, they become increasingly inadequate as data volume, heterogeneity, and longitudinal scope expand. Researchers encounter difficulties in maintaining coherent data organization, ensuring traceability of analytical processes, and enabling reproducible reuse of previously generated results.

These challenges manifest at multiple levels within a research environment. Individual researchers often struggle with inconsistent data organization, limited version control, and poor traceability of data processing steps. At the group level, the absence of a centralized data management approach complicates data sharing, collaborative analysis, and coordinated access to datasets produced by different contributors. Fragmented data handling practices also raise concerns related to access control, data security, and long-term preservation, particularly when working with unpublished or sensitive experimental data.

The situation is further aggravated in experimental domains that rely on multiple measurement technologies and generate heterogeneous datasets. Integrating data from diverse sources typically requires significant manual effort, increasing the likelihood of errors and limiting the scalability and reproducibility of analysis workflows. Without structured support for data integration and management, researchers are often forced to prioritize data handling tasks over scientific interpretation and experimental design.

Therefore, there is a need for a structured data management approach that enables the centralized organization of heterogeneous experimental data, supports secure and controlled data sharing, and facilitates efficient retrieval and analysis. Addressing

this need is essential for improving research efficiency, enabling collaboration, and supporting the long-term usability of experimental data in modern scientific research.

1.3 Objectives and Research Questions

The objective of this study is to design and justify a data management framework capable of integrating heterogeneous experimental research data into a unified and structured system that supports automated ingestion, secure storage, efficient retrieval, and foundational visualization, while reducing manual effort in handling experimental datasets.

This research is guided by the following questions:

RQ-1: How can heterogeneous, instrument-generated experimental data streams be structured, cleaned, and standardized to ensure usability, reproducibility, and long-term accessibility?

RQ-2: Which data management architectures and design principles best support the storage, integration, and retrieval of multi-format experimental data in a collaborative research environment?

1.4 Research Scope and Limitations

The scope of this thesis is defined by the research context of the Photosynthetic Microbes research group at the University of Turku. The system is designed and evaluated within this specific institutional setting, targeting three primary instrument types, the Micro-GC, MIMS, and PBR, whose data handling workflows motivated the research. Of the five data sources associated with the PBR instrument, three are implemented in the current system: the 1000L PhycoFlow (JSON), Meteorological (TXT), and Manual Offline (Excel). The OnePlanet Probe and DAC data sources are identified as future implementation work, as neither instrument is currently available

onsite.

The system is deployed within CSC infrastructure: the database is hosted on CSC Pukki, and the web application is hosted on CSC Pouta. This institutional constraint was a deliberate design choice, ensuring that the system operates within approved, cost-free academic infrastructure. However, it also limits the generalizability of the deployment model to institutions with access to equivalent managed services.

The following are explicitly outside the scope of this thesis:

- **Full FAIR compliance**: The system addresses the Findable, Accessible, and Reusable dimensions of FAIR within its operational scope. Persistent identifier assignment and public metadata publication are identified as future development goals.
- **Semantic interoperability**: The controlled vocabulary used for variable names and units is an internally defined standard, not formally aligned with established scientific ontologies such as ChEBI or UO.
- **User feedback evaluation**: Structured feedback from group researchers on system usability is identified as a planned future evaluation step and is not included in the current evaluation.
- **General-purpose applicability**: While the architectural principles are transferable, the system has been validated only in the photosynthetic research context and would require adaptation for research domains with substantially different data characteristics.

1.5 Thesis Structure

This thesis is organized into eight chapters, each building upon the previous to present a complete research study on research data management for instrument-driven ex-

perimental research. The **Introduction** establishes the motivation for the research, defines the problem, states the research objectives and questions, and outlines the scope and limitations of the study. This is followed by the **Background** chapter, which introduces the core concepts underlying the research, including research data management principles, the FAIR data framework, and the role of automated data pipelines in experimental research settings. The **Literature Review** presents a structured analysis of existing research data management systems, including OMERO, SEEK, Dataverse, and OpenBIS, identifying persistent limitations in automation, interoperability, and usability that motivate the proposed solution. The **Methodology** chapter describes the design science research approach adopted by the study, defines the functional and non-functional requirements derived from the research context, and documents the architectural choices and technology selection that guided the system design. The **System Design and Implementation** chapter presents the proposed data management system in full, covering the system architecture, data model, ingestion pipeline, web-based access and visualization layer, and security considerations. The **Evaluation** chapter assesses the implemented system against the defined requirements through requirements traceability and a comparative analysis against the systems reviewed in the literature. The **Discussion** interprets the evaluation results, answers the research questions, discusses the generalization of the findings to other instrument-driven research settings, and identifies limitations and future development directions. Finally, the **Conclusion** summarizes the principal contributions of the thesis, restates the answers to the research questions, and situates the work within the broader context of open and reproducible research data management practice.

1.6 Declaration of Generative AI Usage

I hereby declare that I have used Claude by Anthropic during the preparation of this thesis. The tool was used to assist with language refinement, grammar correction, academic register improvements, and structural suggestions for written content. All research design, technical implementation, analysis, evaluation, and conclusions presented in this thesis are entirely the original work of the author. No generative AI tool was used to produce or interpret scientific content, conduct the literature review, or generate any part of the implemented system. The author takes full responsibility for the accuracy, integrity, and originality of all content presented in this thesis.

2 Background Concepts

Designing a data management system for experimental research requires grounding in the principles and concepts that such a system must embody. This chapter introduces the fundamental concepts that form the technical and conceptual foundation of this thesis. Rather than providing a general survey, it focuses on the principles that directly shaped the design decisions described in subsequent chapters: how data is structured and classified, how its lifecycle is managed, how quality and governance are maintained, and how metadata enables long-term usability and reproducibility. These concepts establish the vocabulary and criteria used throughout the thesis to evaluate existing systems, justify architectural choices, and assess the proposed solution.

2.1 Fundamentals of Data Management

2.1.1 Introduction to Data Management

Data management is a multi-step process involving the collection, cleaning, transformation, and storage of data to enable accurate analysis and meaningful results. These activities may occur concurrently rather than sequentially and often require coordinated efforts across multiple roles. Establishing clear data management practices prior to data use is essential for maintaining data quality and ensuring that collected data can be reliably interpreted and reused. [1]

Core practices in data management include the use of well-defined data structures, metadata, documentation, and standardized workflows to maintain data quality and support future reuse. Structured formats and consistent naming conventions improve data discoverability and reduce the risk of misinterpretation, particularly in collaborative or long-term projects [2]. The development and application of such practices ensure that data remains interpretable, traceable, and reproducible across different platforms and contexts.

Secure data storage and ethical handling are critical components of effective data management. Measures such as de-identification of sensitive data, use of role-based access controls, and adherence to institutional storage protocols help protect the confidentiality and integrity of stored data [1]. In systems that manage regulated or sensitive information, these practices are essential for ensuring compliance with data protection standards and minimizing the risk of unauthorized access or data breaches.

Many existing systems, however, are limited by rigid structures, insufficient scalability, or lack of interoperability. Contemporary data environments require flexible, modular, and principle-driven architectures that can accommodate diverse data formats and evolving workflows [3]. These limitations highlight the need for purpose-built data management solutions capable of addressing domain-specific requirements while supporting broader goals of automation, integration, and long-term sustainability. In the context of this thesis, these shortcomings directly motivate the design of a lightweight, instrument-aware system for the Photosynthetic Microbes research group, where heterogeneous instrument outputs, manual workflows, and the absence of centralized storage represent precisely the challenges that structured data management is intended to address.

2.1.2 Structured vs. Unstructured Data

Data in modern research and information systems can be broadly categorized into structured, semi-structured, and unstructured forms, depending on how well their organization conforms to predefined models or schemas [4].

Structured data is organized into clearly defined fields, typically stored in relational databases using standardized query languages such as SQL. Its schema defines relationships between entities and attributes, ensuring high data integrity and easy retrieval. Examples include numerical measurements, time-stamped sensor readings, or relational metadata describing experimental parameters [4]. Structured data supports deterministic queries and statistical analysis, making it central to monitoring instruments and capturing consistent, quantitative outputs in laboratory settings.

In contrast, unstructured data lacks a fixed format or schema. It includes textual logs, image files, microscopy data, videos, or free-text observations, data that cannot easily be parsed by conventional relational systems [4]. Such data forms a large portion of modern experimental output, especially in domains that integrate imaging, spectroscopy, and environmental sensors. Between these extremes lies semi-structured data, which uses flexible, self-describing formats such as XML, JSON, or HDF5, enabling schema evolution while preserving contextual metadata.

The increasing heterogeneity of research data has prompted the emergence of hybrid storage solutions capable of handling structured and unstructured forms simultaneously. These include *data lakes* and *data lakehouse architectures*, which allow schema-on-read access, meaning data can be queried without prior structuring [5]. Such architectures are especially advantageous in research contexts where raw instrument outputs and derived analyses coexist.

In experimental sciences, structured data (e.g., quantitative assay results) provides reproducibility and standardization, whereas unstructured data (e.g., instrument

logs, imaging datasets, observational notes) retains critical contextual information. Managing both effectively requires systems that integrate structured databases with flexible object storage, unified through robust metadata frameworks [6], [7].

2.1.3 Data Management Architectures and Design Principles

Data management architectures define the framework through which data is collected, processed, stored, and made accessible within an organization. The architecture directly determines system performance, scalability, interoperability, and the ability to integrate diverse data sources.

Early implementations relied on centralized architectures, such as data warehouses, which consolidate information from multiple sources into a single repository to ensure consistency and control [1]. While these architectures support reliable querying and reporting, they often lack flexibility for environments where data formats or sources change frequently.

To overcome the limitations of centralized architectures, distributed architectures emerged, decentralizing data storage and processing across multiple nodes. This design improves scalability and fault tolerance by enabling parallel execution, though it introduces additional complexity in coordination and system management [3].

In recent years, cloud-based and hybrid architectures have become dominant in both enterprise and research data management. They combine the elasticity of distributed systems with the governance and accessibility of centralized ones [3]. Cloud services enable scalable data ingestion and processing pipelines, supporting collaboration across institutional and geographic boundaries.

Modern developments have introduced metadata-centric architectures, such as data lakehouses, which emphasize interoperability and contextualization over raw storage. The data lakehouse model merges the flexibility of data lakes with the structured governance of warehouses, enabling schema-on-read access and improved

interoperability across heterogeneous research data sources [5].

2.1.4 Data Lifecycle

In research environments, effective lifecycle management extends beyond technical storage; it requires curation, documentation, and metadata enrichment to preserve the scientific context of data. Frameworks such as the Digital Curation Centre (DCC) Curation Lifecycle Model highlight the continuous nature of data stewardship, emphasizing appraisal and re-evaluation to ensure ongoing relevance [8].

Automated lifecycle management tools now integrate data validation, provenance tracking, and workflow logging, providing traceable audit trails that support scientific integrity [9]. Such automation is particularly valuable in experimental setups where data is generated continuously by instruments and must be version-controlled in real time.

A structured lifecycle approach also strengthens compliance with the FAIR principles, ensuring that data remains findable through persistent identifiers, accessible via open interfaces, interoperable through standard formats, and reusable through comprehensive documentation [10].

2.1.5 Data Quality, Governance, and Preservation

Ensuring data quality is a cornerstone of effective data management. High-quality data must be accurate, complete, consistent, timely, and relevant, as these attributes directly influence the reliability of analyses, models, and decisions based on them [1]. Inconsistent or incomplete data not only reduce analytical value but can lead to irreproducible results, misinterpretations, and inefficiencies.

Data governance establishes the policies, roles, and procedures required to manage data as a strategic and trustworthy asset. It defines how data is accessed, shared, and maintained, ensuring compliance with ethical standards and institutional

or legal frameworks [7], [11].

Effective data governance in research settings increasingly relies on automated and metadata-driven processes to ensure transparency, traceability, and reproducibility across distributed and collaborative environments [12], [13]. Embedding governance mechanisms directly into data collection and management workflows reduces reliance on manual enforcement and supports consistent compliance with institutional and ethical standards.

Metadata, data describing data, plays a central role in ensuring interoperability and preserving the experimental context required for correct interpretation. Common metadata standards, such as *Dublin Core* and *ISO 19115*, provide structured templates for description across disciplines [7], [10]. Figure 2.1 illustrates how OpenBIS implements a structured metadata schema linking experimental records to their contextual attributes.

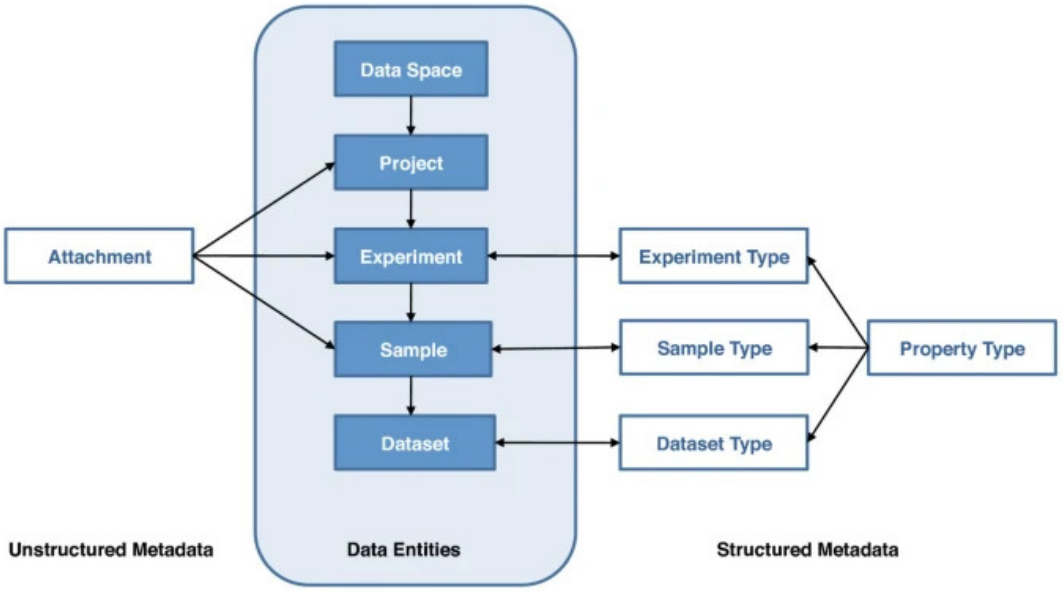

Figure 2.1: Example metadata structure in OpenBIS.

Preservation strategies must ensure that datasets remain accessible and interpretable over time, even as technologies evolve. This involves adopting non-proprietary formats, redundant storage solutions, and periodic data migration to

prevent obsolescence [13], [14]. FAIR-compliant preservation emphasizes metadata richness, persistent identifiers (such as DOIs), and clear licensing to support ongoing reuse [15].

Together, data quality, governance, and preservation form the foundation of sustainable data management. In the context of experimental research, these principles guarantee that instrument-generated data is trustworthy, ethically managed, and reusable across projects and time.

2.2 Research Data Management in Experimental Sciences

2.2.1 Data Management in Research Context

In the context of scientific and experimental research, data management encompasses the systematic processes and infrastructures that ensure data is collected, organized, stored, and preserved in a manner that maintains its integrity, accessibility, and long-term value [7]. Research data management (RDM) emphasizes transparency, reproducibility, and collaboration, contrasting with commercial data management's focus on operational efficiency.

Research data are often complex, multi-format, and dynamically generated, making their management a critical aspect of the research lifecycle. Effective RDM practices begin at the planning stage, where researchers define how data will be collected, annotated, and stored, often formalized through Data Management Plans (DMPs) [2]. These plans outline standards for documentation, storage, sharing, and preservation.

The technical dimension of RDM involves implementing secure, scalable, and interoperable infrastructures for handling large volumes of heterogeneous data. This

includes laboratory information management systems (LIMS), electronic lab notebooks (ELNs), and cloud-based repositories designed to accommodate structured and unstructured datasets, with version control and provenance tracking to maintain reproducibility [1], [7].

Beyond infrastructure, RDM also has a social and organizational dimension [7]. Institutional policies, researcher training, and shared data cultures are key determinants of how effectively RDM is implemented. Challenges such as inconsistent metadata practices, insufficient incentives for data sharing, and limited awareness of metadata standards often hinder adoption [2], [11].

2.2.2 Characteristics of Research Data

Research data differ significantly from business or transactional data in terms of complexity, structure, and purpose. They are often heterogeneous, encompassing numerical measurements, images, textual logs, and sensor outputs from diverse instruments and methodologies [7].

A defining feature of research data is **provenance**, the documentation of data origin, collection methods, and processing steps. Provenance metadata ensure traceability and reproducibility, allowing results to be verified independently [2]. Equally critical is the contextual dependency of research data; their interpretation depends on experimental conditions, instrument quality and settings, calibration procedures, measurement precision requirements, and specific experimental protocols, all of which must be preserved alongside the data itself [7].

Research data also require careful version management. Raw data collected directly from instruments should remain unchanged; any subsequent cleaning, normalization, or processing steps should be stored as separately documented derived versions rather than replacing the original records [3]. This distinction between raw and processed data is essential for reproducibility, as it ensures that the original

measurements remain available for independent verification and reanalysis. In some experimental environments, certain instruments produce data at high frequency, creating additional challenges in scalability and automation [6], though this characteristic varies considerably across instrument types and experimental designs.

Finally, research data are often collaborative and may need to remain usable over long periods. Without appropriate metadata and standardization, datasets risk becoming isolated and unusable over time [7]. Effective RDM frameworks, therefore, emphasize not only storage and retrieval but also context preservation and long-term accessibility.

2.2.3 FAIR Data Principles in Research

The **FAIR Data Principles** — Findable, Accessible, Interoperable, and Reusable — constitute an internationally recognized framework for ensuring the quality and sustainability of research data management [10]. Introduced by Wilkinson et al. in 2016, these principles were designed to enhance data transparency, reproducibility, and longevity across all scientific disciplines.

A dataset is considered **Findable** when it is described with rich metadata and assigned a persistent identifier, such as a DOI or Handle. **Accessible** data should be retrievable via standardized communication protocols such as HTTPS or API endpoints. **Interoperability** emphasizes the use of shared formats, vocabularies, and ontologies to enable seamless integration with other datasets and analytical tools. Finally, **Reusability** ensures that data are well-documented, licensed appropriately, and accompanied by sufficient provenance information [10].

Applying FAIR in experimental sciences remains challenging due to the heterogeneous nature of data, domain-specific standards, and legacy infrastructure [16]. Despite these challenges, FAIR adoption enhances collaboration, reproducibility, and cross-disciplinary integration [17].

2.3 Summary

This chapter has outlined the fundamental concepts that underpin effective data management and the development of research data management systems. It established that robust data practices rely on well-structured architectures, standardized lifecycles, and clear governance frameworks that ensure quality, security, and long-term accessibility. The distinction between structured and unstructured data highlights the need for flexible systems capable of handling heterogeneous information, while principles such as provenance, documentation, and metadata reinforce the importance of reproducibility and traceability in research.

The FAIR principles were introduced as a foundational guideline for ensuring that data remain discoverable, interoperable, and reusable across domains and over time. Together, these concepts form the theoretical and technical basis for evaluating existing research data management systems in the following chapter.

3 Previous Research on Scientific Data Management

The background concepts established in Chapter 2 provide the theoretical foundation for evaluating how existing systems have attempted to address research data management challenges in practice. This chapter reviews the academic literature and existing systems related to scientific and experimental data management, with the specific purpose of identifying what has been built, where those systems fall short for instrument-driven experimental research settings, and what design implications follow for the system proposed in this thesis. A structured narrative review approach with critical analysis and integrative synthesis is adopted. Sources were analyzed thematically to identify key design principles, persistent challenges, and gaps in existing systems. The findings from this review directly informed the requirements, architecture, and technology choices described in Chapter 4.

3.1 Research on Scientific Data Management

Scientific Data Management (SDM) refers to the systematic processes and infrastructures that support the collection, organization, curation, preservation, and sharing of research data throughout its lifecycle [7]. With the rapid expansion of data-driven science, research outputs have become increasingly diverse in type and scale, encompassing structured, unstructured, and streaming data from multiple instruments.

This growth in volume, velocity, and variety of scientific data has underscored the need for structured and sustainable management frameworks [2].

Modern SDM extends far beyond data storage. It incorporates metadata standardization, provenance tracking, workflow documentation, and adherence to open-science practices such as the FAIR principles [10]. Studies have shown that FAIR-aligned practices significantly improve data reuse and interdisciplinary collaboration [18].

However, effective SDM also requires integration of technical, social, and organizational dimensions. Borgman [7] emphasizes that technological solutions alone are insufficient without aligning institutional policies and research cultures toward active data stewardship, and that long-term data sustainability equally depends on governance structures, researcher engagement, and consistent adherence to metadata standards.

Despite progress in digital infrastructure, challenges persist. These include heterogeneous data formats, insufficient interoperability, and metadata quality issues, all of which hinder reproducibility and efficient reuse [19]. Moreover, many researchers cite the lack of time, training, and institutional support as barriers to proper data documentation [18].

3.2 Existing Research Data Management Systems

Over the past two decades, numerous Research Data Management (RDM) systems have been developed to address the increasing complexity and volume of scientific data. These systems aim to enable reproducibility, facilitate collaboration, and ensure long-term data preservation across disciplines. Among the most prominent open-source systems are **OMERO**, **SEEK**, **Dataverse**, and **OpenBIS**, each developed to meet specific community needs while adhering to shared principles of data transparency and interoperability [20].

3.2.1 Overview of Existing Systems

Although these platforms differ in implementation and domain scope, they collectively illustrate the evolution of RDM practices from simple repository-based storage to integrated, metadata-driven frameworks.

Table 3.1 summarizes their primary domains, core functionalities, and limitations.

Table 3.1: Comparison of existing RDM systems.

System	Domain	Core Features	Limitations
OMERO	Microscopy and imaging	Model-driven data management, metadata annotation, role-based access, visualization tools	Not optimized for continuous sensor data
SEEK	Systems Biology	Workflow integration, ontology-based metadata, semantic data linking	Complex configuration
Dataverse	General research	Dataset publishing, DOI-based citation, access control	Requires manual metadata curation
OpenBIS	Life sciences and Chemistry	Sample tracking, metadata linkage, LIMS integration	Complex setup, steep learning curve

OMERO OMERO (Open Microscopy Environment Remote Objects) is a widely adopted open-source platform for managing microscopy and imaging data [21]. It provides a model-driven data architecture that supports metadata storage, access control, and interactive visualization of large multidimensional datasets. However, OMERO is highly specialized for image data and less suited for heterogeneous datasets or continuous sensor outputs typical in experimental settings [22].

SEEK SEEK is designed primarily for systems biology, where reproducibility and workflow documentation are essential. It allows researchers to link experimental data, computational models, and protocols within a semantic framework [23]. Despite its strengths, SEEK presents notable limitations in practice: its reliance on controlled vocabularies and immutable sample records makes it difficult to adapt to ongoing, evolving experimental workflows, and its visualization capabilities remain limited [18], [23]. Extensions such as NExtSEEK have addressed some of these constraints by enabling mutable metadata and active data management [24], yet the underlying configuration complexity continues to pose barriers for smaller research groups.

Dataverse Developed by Harvard University's Institute for Quantitative Social Science, Dataverse provides a general-purpose framework for publishing, citing, and preserving datasets across disciplines. Each dataset is assigned a DOI, promoting persistent identification and citation. While its repository-oriented approach enhances preservation and discoverability, Dataverse is not designed for dynamic or real-time data and lacks direct pipelines for ingesting data from laboratory instruments.

OpenBIS OpenBIS (Open Biological Information System) is a modular platform designed for managing complex datasets in biological and chemical research [25]. It links raw data to project metadata and integrates seamlessly with laboratory information management systems (LIMS). However, OpenBIS requires extensive configuration and technical expertise for deployment and maintenance [26]. Its layered architecture, illustrated in Figure 3.1, reflects the platform's emphasis on modularity, though this same complexity contributes to its steep learning curve for smaller research teams.

Taken together, the above assessments of available open-source RDMs indicate that current solutions are still far from the requirements of modern experimental research. Most existing platforms are optimized for static or domain-specific data

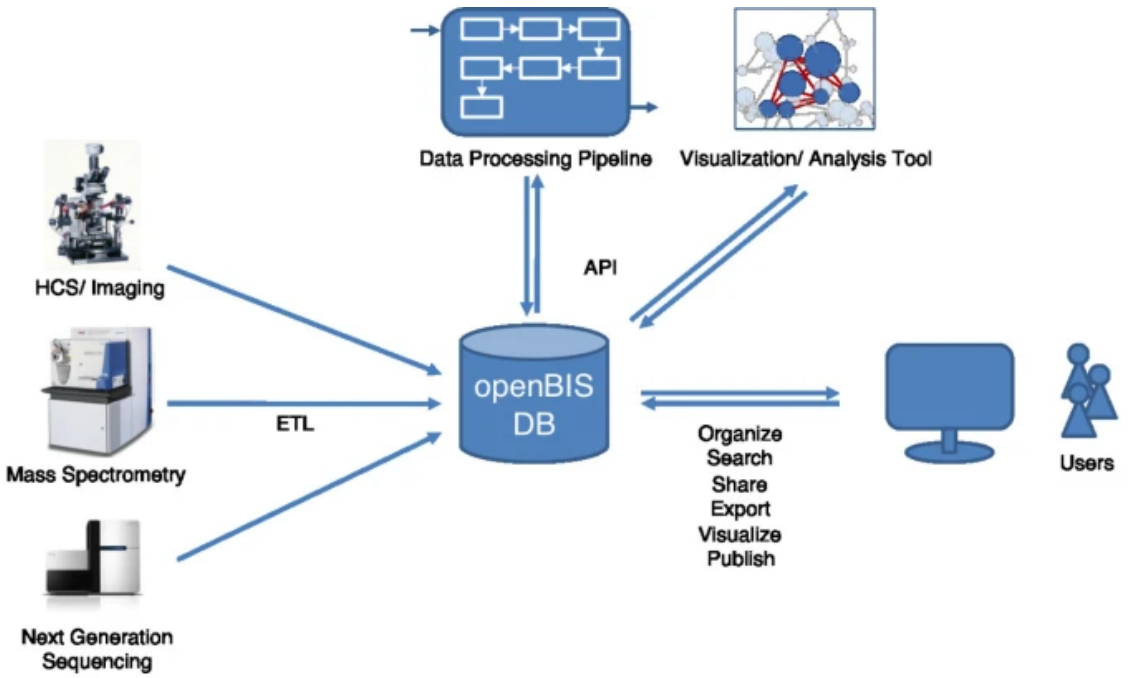

Figure 3.1: OpenBIS architecture overview.

rather than heterogeneous data streams typical of experimental instruments. Additionally, the requirement for manual metadata entry introduces inconsistencies, undermining reproducibility [20].

3.3 Case Studies in Experimental and Biological Data Management

Examining how RDM systems have been deployed in real experimental settings reveals practical limitations that are not always visible from system documentation alone. The following case studies illustrate the recurring difficulties encountered when applying existing platforms to instrument-driven biological and experimental research.

Mittal et al. [18] examined data management practices within a large collaborative research center and identified recurring issues such as fragmented data storage, lack of standardized metadata, and limited interoperability among laboratory instruments

and repositories. Their study emphasized that effective integration of heterogeneous data requires well-defined metadata schemas and automated data ingestion mechanisms. A key finding was that successful implementation depends not only on robust technical infrastructure but also on organizational and cultural alignment toward open-science practices. Importantly, even in a well-resourced center, the absence of automated ingestion pipelines resulted in data quality inconsistencies that compromised reproducibility across research groups.

Bauch et al. [27] presented the implementation of OpenBIS within molecular biology laboratories, where modular data models enabled the linking of raw experimental data to project metadata and analytical outcomes. While the platform provided a structured foundation for data organization, the study noted that significant administrative configuration and maintenance efforts were required before the system became operational. This overhead limited accessibility for smaller research teams that lacked dedicated IT support, and the authors concluded that the complexity of OpenBIS made it poorly suited to groups without a technical administrator. This finding directly parallels the situation of the Photosynthetic Microbes research group, where researchers are domain scientists rather than software engineers, and where a system requiring specialist administration would face adoption barriers from the outset.

Taken together, these cases illustrate that the gap between available RDM platforms and the practical needs of smaller, instrument-driven research groups is not merely a feature gap but a usability and deployment gap. Both studies highlight automated ingestion and low administrative overhead as the most consequential unmet requirements, which directly informed the design priorities of the system proposed in this thesis.

3.4 Limitations and Gaps in Existing Approaches

Although numerous RDMS have been developed, their practical deployment within instrument-driven experimental environments reveals persistent structural limitations. These limitations reflect a misalignment between system design assumptions and the realities of heterogeneous experimental workflows.

One of the most prevalent limitations is limited adaptability to heterogeneous experimental data. Systems such as OMERO, SEEK, Dataverse, and OpenBIS were each developed with specific types of data or research workflows in mind [21], [23], [25], [27]. Most current systems struggle to automatically capture, integrate, and process diverse data sources in real time, relying instead on manual uploads or batch imports. This lack of automation increases the potential for human error and undermines reproducibility [2], [28].

Usability and accessibility also present ongoing challenges. While systems like OpenBIS and SEEK provide robust backend architectures, their interfaces and configuration requirements often demand considerable technical expertise [25]. Smaller research groups without dedicated IT support find it difficult to deploy or maintain such systems effectively [29].

Interoperability represents another critical gap. Despite growing emphasis on the FAIR principles, data exchange between systems remains limited due to inconsistent metadata standards and incompatible ontologies [10]. Domain-specific vocabularies and proprietary data formats often prevent seamless integration even within the same institution [7].

From an organizational perspective, sustaining long-term data curation poses an equally significant challenge. Studies highlight that there remains a lack of institutional mechanisms for supporting continuous curation, training, and maintenance of RDM infrastructures [2], [7].

3.4.1 Implications for Designing a New Data Management System

Findings from existing research and system implementations reveal that the design of an effective DMS for experimental research must directly address the shortcomings observed in current RDMS. A central implication is the necessity for **automation and real-time data capture**. Automated pipelines for ingesting data directly from laboratory instruments, coupled with mechanisms for immediate metadata annotation, can significantly enhance data quality and ensure that information remains synchronized across storage layers [18].

Another important consideration is **architectural modularity and scalability**. A modular DMS design based on interoperable components and open APIs would enable the integration of new data types and instruments without requiring complete system reconfiguration [21].

Usability and accessibility also emerge as key factors influencing system adoption. A researcher-centric design philosophy that prioritizes intuitive interfaces, clear workflows, and minimal administrative overhead can promote consistent data entry [25].

Finally, all technical and usability considerations must align with FAIR and open-science principles to ensure long-term data value [10].

Table 3.2 synthesizes the key limitations identified in existing systems and translates them into concrete design requirements for the proposed system.

3.5 Summary of Findings

The reviewed literature demonstrates a steady evolution in scientific data management from traditional storage-focused models toward integrated, interoperable, and FAIR-oriented frameworks. Existing RDM systems, such as OMERO, SEEK, Dataverse, and

Table 3.2: Limitations in existing RDMS and implications for new system design.

Identified Limitation	**Supporting Literature**	**Implication for Design**
Manual data ingestion and post-hoc metadata entry	[2], [18]	Implement automated real-time ingestion and metadata capture
Domain-specific rigidity (OMERO, SEEK)	[21], [23]	Design modular, extensible architecture with open APIs
Complex configuration and steep learning curves	[25], [27]	Prioritize researcher-centric usability and minimal setup overhead
Interoperability limitations	[7], [10]	Adopt standardized metadata schemas and persistent identifiers
Lack of sustainability mechanisms	[7]	Embed governance, versioning, and long-term preservation policies

OpenBIS, each contribute valuable concepts but also reveal fundamental shortcomings, including limited automation, lack of scalability for diverse data types, and high administrative overhead.

The literature also highlights the broader socio-technical dimensions of research data management. Effective implementation depends not only on technical infrastructure but also on institutional governance, user engagement, and adherence to open-science practices [7].

Overall, the synthesis of findings demonstrates that while current research data management systems provide valuable infrastructure components, none comprehensively address the combined requirements of automated instrument-level data ingestion, heterogeneous data integration, and researcher-centric usability with low administrative overhead within dynamic experimental environments. Although platforms such as OpenBIS offer modular architectures, their complexity and configuration demands limit their practical accessibility for smaller, instrument-driven

research groups.

4 Research Design and Methodology

4.1 Research Approach

This thesis adopts a design science research (DSR) methodology, which is well-suited to information systems research that aims to produce a purposeful artifact, in this case, a functional data management system, as a primary contribution, accompanied by a rigorous account of the design rationale grounded in domain-specific requirements and a clearly defined real-world problem [30], [31]. DSR is characterized by the iterative construction, evaluation, and refinement of artifacts, and has been widely applied in the design of scientific information systems and data management frameworks [32].

The study proceeds through three interconnected phases. The first phase involves a structured review of existing research data management systems and related literature, the results of which are presented in Chapter 3. The second phase involves the design and implementation of a data management system (DMS), grounded in those requirements and informed by established data management principles. The third phase constitutes an evaluation, in which the implemented system is assessed against the identified requirements and compared with existing solutions.

The research is primarily qualitative and constructive in nature. Design choices are justified through explicit reasoning grounded in the literature reviewed in Chapter 3 and the operational requirements identified through direct engagement with the Photosynthetic Microbes research group.

4.2 Experimental Research Context

The Photosynthetic Microbes group conducts its experiments using instruments that assess culture growth and cell fitness in microalgae and cyanobacteria by monitoring optical density, photosynthetic activity, and associated gas exchange. These instruments include the Micro Gas Chromatograph (Micro-GC), the Membrane Inlet Mass Spectrometry (MIMS) setup, and a microprocessor-controlled photobioreactor (PBR), which are the main focus of this thesis. Data produced from each of these devices provide valuable insights into photosynthetic performance and productivity of microbial cultures under batch, semi-continuous, and continuous cultivation conditions.

4.2.1 µGC — Micro Gas Chromatography

The Micro Gas Chromatograph (Micro-GC) is a miniaturized analytical instrument designed to separate and quantify gaseous components within a sample. It operates on the same principle as conventional gas chromatography but integrates key components on a microfabricated platform [33].

Within the Photosynthetic Microbes group, the Micro-GC is used to analyze the composition of gases such as hydrogen (H_2), oxygen (O_2), nitrogen (N_2), and carbon dioxide (CO_2) in experimental headspaces or output streams. Each measurement run produces a proprietary `.sirslt` archive — a 7-Zip container holding instrument-specific files including raw measurement data (`.rx`), method configuration (`.amx`), calibration data (`.sqx`), and an ACAML-formatted analytical record (`.acaml`).

For downstream processing, summarized results are exported from the instrument software as CSV files containing retention times, peak areas, and calibrated concentrations for each detected gas species. It is these CSV exports that the ingestion pipeline reads and parses. The preservation of the raw `.sirslt` archives is identified as a future consideration for long-term data provenance.

4.2.2 MIMS — Membrane Inlet Mass Spectrometry

The time-resolved Membrane Inlet Mass Spectrometry (MIMS) is a highly sensitive analytical instrument used to monitor gas exchange processes in photosynthetic and respiratory systems [34]. Within the Photosynthetic Microbes group, MIMS serves as a primary instrument for studying real-time gas exchange in cyanobacterial and microalgal cultures under varying experimental conditions. The real advantage of MIMS is its ability to measure gas isotopologues, enabling the separation of gross and net photosynthesis and the simultaneous, high-sensitivity quantification of respiratory and photo-respiratory fluxes.

Each measurement produces time-series datasets containing ion-current intensities, calibration parameters, and experimental data [34]. In the group's experimental workflows, individual MIMS measurement sessions are typically short-term, lasting between 20 and 40 minutes, and produce structured CSV output that requires organized storage and consistent data handling practices.

4.2.3 PBR — Photobioreactor

A photobioreactor (PBR) is a closed, illuminated vessel designed to cultivate photosynthetic microorganisms such as microalgae and cyanobacteria under controlled environmental conditions [35]. Within the Photosynthetic Microbes group, PBRs play a central role in developing sustainable microalgae production platforms. The group operates a 1 m^3 photobioreactor in the Ruissalo greenhouse, Turku, enabling large-scale cultivation of Nordic microalgae using recycled nutrient-rich drain water.

During operation, the PBR continuously monitors key parameters such as temperature, pH, dissolved oxygen, gas flow rates, and light intensity. These variables are logged as time-series data that describe both environmental conditions and culture responses over extended experimental periods.

4.3 Data Sources and Collection

The experimental data used in this study is derived from multiple instruments and systems associated with photosynthetic research activities. The data sources vary significantly in structure, acquisition frequency, and accessibility, reflecting the heterogeneous nature of research data encountered in laboratory and field-based experimental setups. This section focuses on the file formats, data access methods, acquisition frequency, and automation potential of each source. The experimental role of each instrument within the group's research workflows is described in Section 4.2.

The three main instruments providing data for this study are the **Micro-Gas Chromatograph (Micro-GC)**, the **Membrane Inlet Mass Spectrometer (MIMS)**, and the **Photobioreactor (PBR)** setup. Each of these systems produces structured or semi-structured datasets, typically in machine-readable formats such as CSV or JSON.

The **Micro-GC** natively produces a proprietary `.sirslt` archive per run — a 7-Zip container holding instrument-specific files including `.rx` (raw measurements), `.amx` (method), `.sqx` (calibration), and `.acaml` (ACAML analytical record). As these formats are instrument-specific and not directly parseable by standard tools, summarized results are exported as CSV files for ingestion. At present, data retrieval requires manual access to the local storage of the instrument's controlling computer.

The **MIMS** system generates structured CSV files containing gas exchange time-series data, well-organized and timestamped per measurement session. The system's standardized output structure allows straightforward incorporation into automated data pipelines.

The **Photobioreactor (PBR)** setup represents the most complex data source, encompassing multiple interconnected subsystems. Data from the PBR system are produced as JSON files at regular intervals (approximately every 30 seconds), capturing up to 25 environmental and operational variables.

In addition to the PBR's core dataset, several auxiliary data sources contribute essential contextual information:

- **Meteorological data**, collected from a dedicated Windows PC in the on-site greenhouse, include approximately four variables measured every four minutes. The PC operates without internet or UTU network connectivity, making full automation infeasible under current constraints; data is therefore collected manually on a monthly basis via USB transfer.

- **Manual or offline measurements**, such as optical density and culture observations, are initially recorded on paper and subsequently digitized into Excel files. While the data entry step itself cannot be automated, downstream processing steps, including calculation of averages, standard deviations, and visualization, can be automated once the Excel files are uploaded to the database.

- **OnePlanet Probe Box** data, comprising 13 variables logged every ten minutes as CSV exports, and its data resides on an external cloud server.

- **Direct Air Capture (DAC)** system data, comprising approximately 93 variables sampled every five seconds as Excel exports. The DAC system is used to capture CO_2 from ambient air and is associated with PBR experiments where CO_2 availability and delivery are relevant to microalgae cultivation performance. Both the OnePlanet Probe and DAC data sources are scoped as future implementation work, as neither instrument is currently available onsite.

Table 4.1 summarizes all instrument data sources.

Table 4.1: Summary of instrument data sources in the Photosynthetic Microbes group.

Instrument	Format	Frequency	Variables	Current Access	Automation
Micro-GC	`.sirslt` (CSV export)	Per run	~Multiple	Local storage; manual copy	High
MIMS	CSV	Per session (20–40 min)	Structured data	Manual transfer	High
PBR (Online)	JSON	Every 30 s	~25	Remote via AnyDesk	High
Meteorological	CSV	Every 4 min	~4	Manual via USB	Low
Offline	Excel	As entered	Variable	Manual entry	Medium
OnePlanet Probe	CSV	Every 10 min	~13	External cloud server	Medium
DAC	Excel	Every 5 s	~93	External source	Medium

4.4 Requirements Gathering and System Specification

System requirements were derived from two complementary sources: a synthesis of the limitations identified in existing research data management systems (as reviewed in Chapter 3), and a contextual analysis of the data management practices currently in use within the Photosynthetic Microbes research group at the University of Turku.

The analysis revealed several areas where current practices could be strengthened: the absence of a centralized repository accessible to all authorized group members, inconsistent naming conventions across instrument outputs, a lack of structured metadata linking datasets to their experimental context, and an over-reliance on manual file transfers that introduce transcription risk and version conflicts.

The following **functional requirements** were formulated:

- **FR1**: The system shall support the ingestion of heterogeneous data files from at least three primary instrument types: Micro-GC (CSV), MIMS (CSV), and PBR (JSON).
- **FR2**: The system shall apply standardized naming conventions and contextual annotations to all ingested files prior to storage.
- **FR3**: The system shall store data in a centralized, cloud-accessible repository accessible to authorized project members through role-based authentication.
- **FR4**: The system shall provide a mechanism for browsing and querying stored datasets by experiment, instrument type, date range, and researcher.
- **FR5**: The system shall provide a visualization interface capable of rendering measurement data from stored datasets.
- **FR6**: The system shall maintain traceability of data origin, collection date, and any applied transformations.

The following **non-functional requirements** were also identified:

- **NFR1**: The system shall be deployable within the University of Turku's existing CSC infrastructure without requiring external proprietary services.
- **NFR2**: The system shall reduce manual effort in routine data handling through automated ingestion pipelines where feasible.
- **NFR3**: The system shall be extensible to support additional instrument types and data sources, with new instruments requiring new ingestion scripts but no changes to the core database schema.

- **NFR4**: The system shall comply with the FAIR data principles to the extent feasible within its operational scope [10].
- **NFR5**: The system shall implement role-based access controls to protect unpublished experimental data.
- **NFR6**: The system shall provide structured logging and error reporting for all ingestion events to support operational monitoring and auditability.

These requirements collectively inform the system design presented in Chapter 5 and serve as the evaluation criteria applied in Chapter 6.

4.5 System Architecture and Data Flow Planning

The architecture of the proposed DMS is designed around a four-layer model comprising a data source layer, a data ingestion layer, a centralized storage layer, and a data access and visualization layer. This layered approach reflects established principles in data management architecture, which emphasize the separation of concerns between data collection, storage, and consumption to improve maintainability and scalability [36].

The data flow begins at the instrument level, where raw data files are generated in instrument-specific formats, including proprietary archives (Micro-GC), JSON (PBR), CSV (MIMS), tab-separated TXT (Meteorological), and Excel (Manual Offline). A Python-based ingestion pipeline, driven by an event-based file system watcher, monitors designated source directories on instrument-connected computers and triggers processing automatically when new files are detected. Upon detecting new files, the pipeline performs format validation, parses measurement values into normalized rows, and loads the structured data directly into the PostgreSQL database hosted on CSC Pukki [37]. Each ingestion event is logged in the `processed_files`

table, which records the source file path, file hash, and row count to support deduplication and auditability.

At the storage layer, parsed measurements are organized relationally within a centralized cloud-hosted database, linked to their experimental context through a hierarchical data structure. The access and visualization layer provides both an interactive web interface for dataset discovery and a programmatic API for external tool integration.

4.6 Tools, Platforms, and Technology Considerations

The selection of tools and platforms was guided by two primary principles: institutional compatibility, meaning that chosen technologies should integrate cleanly with the University of Turku's existing CSC infrastructure and not introduce external service dependencies, and open-source availability, meaning that no component of the system should require a commercial license that would limit reproducibility or long-term maintainability.

Database. The institutional context made the database selection straightforward. CSC Pukki provides a managed PostgreSQL Database-as-a-Service that is free for Finnish universities and fully integrated with the CSC identity management system [37]. Alternative self-hosted options, such as running a PostgreSQL instance on a virtual machine, were considered but rejected, as they would require manual server administration and backup management. PostgreSQL was also the technically appropriate choice given the relational, schema-driven nature of the instrument data and the need for foreign key relationships between experiments, instruments, and measurement records [3].

Ingestion pipeline. Python was selected as the implementation language due to

its dominant position in the scientific computing ecosystem and its extensive library support for the file formats encountered in this project. The `pandas` library handles CSV and Excel parsing; `psycopg2` and `SQLAlchemy` manage database connectivity and ORM-based schema definition; and the `watchdog` library provides the event-driven file system monitoring that underpins the automated ingestion trigger [38]. Alternative languages were not formally evaluated, as Python's library ecosystem and widespread adoption in research computing made it the clear choice within the institutional context.

Visualization. During development, two established open-source visualization platforms were evaluated as candidates for the dashboard layer: Grafana and Apache Superset. Grafana was configured and connected to the live PostgreSQL database, and a set of time-series panels was built and tested against ingested data. Apache Superset was similarly installed and configured. Both were ultimately rejected: Grafana is primarily designed for infrastructure monitoring and requires a separately running service with its own authentication layer, fragmenting the user experience; Apache Superset, while more analytics-oriented, has a complex installation and configuration process unsuitable for a small development team without dedicated infrastructure. Recharts was selected instead because it integrates natively into the React single-page application, delivering visualization and data browsing within a single unified interface with no additional service dependencies.

Web application. A decoupled architecture was chosen in which a FastAPI backend serves a REST API consumed by a React SPA built with Vite. This separation allows the frontend and backend to be developed, tested, and deployed independently, and exposes a REST API that makes system data programmatically accessible to external tools without requiring direct database credentials.

Table 4.2 summarizes the principal technology choices and their justifications.

4.7 Rationale Behind Architectural Choices

Three principal architectural choices shape the proposed system: a layered architecture separating ingestion, storage, and access; a managed relational database as the central storage backend; and an automated, event-driven ingestion pipeline. Each of these choices reflects a deliberate alignment between the requirements of the Photosynthetic Microbes research environment and the design implications identified in the literature review.

Layered architecture. The decision to separate the system into four distinct layers, data sources, ingestion, storage, and access, is motivated by the need for modularity and maintainability. As highlighted in Section 3.4, one of the persistent shortcomings of existing RDM systems such as OpenBIS and OMERO is their tightly coupled architectures, which make it difficult to extend or replace individual components without disrupting the system as a whole [21], [25]. A layered design ensures that adding a new instrument type requires a new ingestion script but does not necessitate changes to the storage schema or the web application. It should be noted, however, that adding a new instrument may still require updates to metadata fields, parsing logic, validation rules, or visualization configuration, depending on the characteristics of the new data source.

Managed relational database. The choice of a managed relational database service (PostgreSQL on CSC Pukki) is justified by both the structured nature of the instrument data and the institutional infrastructure available at the University of Turku [3], [37]. The relational model supports the hierarchical data structure that links experimental context to individual measurements: groups contain instruments, instruments produce data sources, data sources are associated with experiments, and experiments link to measurement records. This hierarchy constitutes the system's data model and is implemented through foreign key relationships between the corresponding tables. The use of a JSONB column for experiment-level metadata

allows flexible, schema-on-write capture of contextual attributes, organism identity, cultivation mode, and experimental conditions, without requiring schema changes for each new attribute [39].

Automated ingestion pipeline. Automation of the ingestion pipeline is justified by the documented risks of manual data handling in experimental research environments [2], [18]. Manual file transfers introduce transcription errors, version conflicts, and gaps in provenance tracing. The ingestion layer follows the ETL (Extract, Transform, Load) pattern: instrument files are extracted from monitored directories, transformed into normalized measurement rows with controlled vocabulary applied, and loaded into the central database. An event-driven pipeline that monitors source directories and triggers ingestion automatically when new files are detected substantially reduces the manual handling risks described above, though it does not eliminate them, as network interruptions, instrument computer unavailability, or format changes may still require manual intervention.

Together, these three architectural choices constitute a coherent response to the requirements identified in Section 4.4 and the design implications articulated in Section 3.4.

Table 4.2: Technology choices for the proposed DMS.

Component	**Technology**	**Justification**
Database	CSC Pukki (PostgreSQL)	Institutionally supported DBaaS; free for Finnish universities; enables structured, queryable relational storage; self-hosted alternatives rejected due to administration overhead [37]
Ingestion Pipeline	Python (pandas, psycopg2, SQLAlchemy)	Widely adopted in scientific computing; extensive support for CSV/JSON/Excel parsing and PostgreSQL interaction [38]
Visualization	Recharts (React library)	Natively integrated into the React SPA; eliminates need for a separate visualization service; Grafana and Superset evaluated and rejected due to infrastructure overhead
Web Dashboard	FastAPI (backend) + React/Vite (frontend)	Decoupled architecture enables independent development and deployment; REST API enables programmatic data access independent of the web interface
Access Control	CSC Pukki user accounts + application-level roles	Integrated with institutional infrastructure; `is_admin` flag in `users` table governs dashboard permissions [37]
Database Client	DBeaver	Open-source universal database client used for schema inspection, query execution, and data verification during development and testing

5 System Design and Implementation

5.1 System Overview

The proposed data management system (DMS) is designed to provide the Photosynthetic Microbes research group at the University of Turku with a centralized, structured, and accessible infrastructure for managing experimental data generated by the Micro-GC, MIMS, and PBR instruments. These three instrument types represent the initial implementation and validation scope of the system. By design, the architecture is extensible: new instruments and data sources can be integrated by adding ingestion scripts and registering new data source records, without changes to the core database schema or shared pipeline infrastructure. This extensibility is a deliberate architectural property, not a future afterthought, and is described in detail in Section 5.8.

The system addresses the fragmented and manual data handling practices described in Chapter 4 by introducing automated ingestion pipelines, a cloud-based storage backend, a hierarchical data schema, and researcher-facing visualization and browsing interfaces.

The system architecture follows a four-layer model, as illustrated in Figure 5.1. The data source layer encompasses the instruments and their native file formats. The data ingestion layer comprises Python-based scripts responsible for detecting, validating, transforming, and loading new data files from instrument-local storage

into the central database. The storage layer consists of CSC Pukki, a managed PostgreSQL Database-as-a-Service provided by CSC – IT Center for Science, Finland [37], in which parsed measurements and experiment metadata are stored relationally. The access and visualization layer includes a React-based web application for browsing and querying stored experimental datasets, file upload, and interactive visualization, and a FastAPI REST API that serves as the communication layer between the frontend and the database.

The system is designed to be deployed incrementally. The ingestion pipeline and storage layer constitute the minimum viable system, enabling centralized data collection from all three primary instruments. The architecture is intentionally modular, allowing individual components to be extended or replaced without disrupting the overall data flow [36].

5.2 Implementation Plan

The implementation of the system has been organized into four sequential phases.

Phase 1 - Database Infrastructure Setup: Establish the CSC Pukki PostgreSQL database instance, define the schema including all reference, experiment, and measurement tables, and configure user-level access permissions. This phase establishes the relational storage foundation that all subsequent ingestion operations write into.

Phase 2 - Ingestion Pipeline Development: Develop and test Python ingestion scripts for each primary instrument type (Micro-GC, MIMS, PBR). Each script handles format-specific parsing, data validation, measurement normalization, and insertion into the appropriate measurement table in PostgreSQL via CSC Pukki. A file system watcher triggers ingestion automatically when new files are detected in monitored source directories.

Phase 3 - Metadata and Schema Refinement: Finalize the

`experiment_metadata` JSONB schema within the `experiments` table. Integrate controlled vocabulary enforcement into the ingestion scripts for `variable_name` and `unit` fields. Verify foreign key integrity across all measurement tables.

Phase 4 - Visualization and Web Dashboard Deployment: Deploy the FastAPI backend and the React web application for dataset browsing, file upload, and interactive visualization using the Recharts library. Integrate the visualization layer with the REST API to enable dynamic chart rendering from ingested measurement data.

This phased approach allows the system to deliver immediate value, centralized, standardized data storage, while deferring the more complex visualization components to later stages.

5.3 Data Architecture and Workflow Design

The data architecture of the proposed system is built around a hybrid storage model that separates the concerns of raw file handling from structured data persistence. Instrument outputs enter the system as files in their native formats and are consumed by the ingestion layer, which is responsible for extracting, validating, and transforming their contents before writing normalized records into the central relational database. The raw source files are not stored in the database in their original form; instead, the pipeline distills each file into a set of structured measurement rows, preserving only the unparsed record in a JSONB provenance column alongside the parsed values. This design keeps the storage layer clean and queryable while retaining full traceability back to the original instrument output.

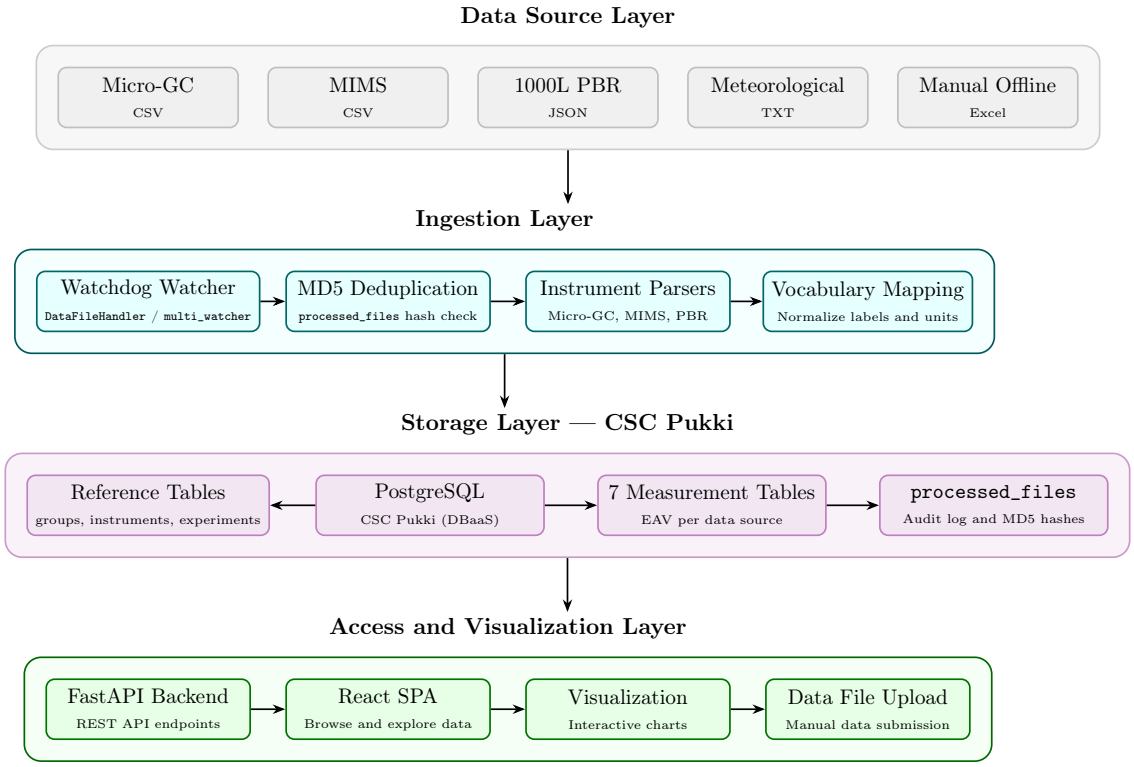

Figure 5.1: Four-layer system architecture of the proposed DMS. The data source layer encompasses the three instruments and their native file formats. The ingestion layer comprises the `watchdog`-based file watcher and `multi_watcher` orchestrator, MD5 hash deduplication, instrument-specific parsers, and controlled vocabulary mapping. The storage layer is the PostgreSQL database hosted on CSC Pukki, consisting of reference and experiment tables, seven EAV measurement tables, and the `processed_files` audit log. The access and visualization layer, deployed on CSC Pouta, provides a FastAPI REST API, a React single-page application with integrated time-series visualization, and a file upload feature for manual data submission.

5.3.1 Conceptual Data Model

The conceptual data model is organized around five layers of abstraction, progressing from organizational context down to individual measurements. These layers are realized as relational tables within the central PostgreSQL database schema.

At the highest level, the `groups` table defines the organizational unit within which all research activity takes place. The `instruments` table enumerates the physical measurement instruments (MicroGC, MIMS, PBR). The `data_sources` table provides a finer-grained classification of data origin, distinguishing between the

multiple data streams a single instrument platform may produce. Seven data sources are registered in the system: MicroGC_Main, MIMS_Main, 1000L_PhycoFlow, Meteorological, Manual_Offline, OnePlanet_Probe, and DAC. Of these, three PBR data sources (1000L_PhycoFlow, Meteorological, and Manual_Offline) are currently implemented alongside MicroGC_Main and MIMS_Main; OnePlanet_Probe and DAC are identified as future work. Their mapping to instrument types, file formats, and target measurement tables is shown in Figure 5.2.

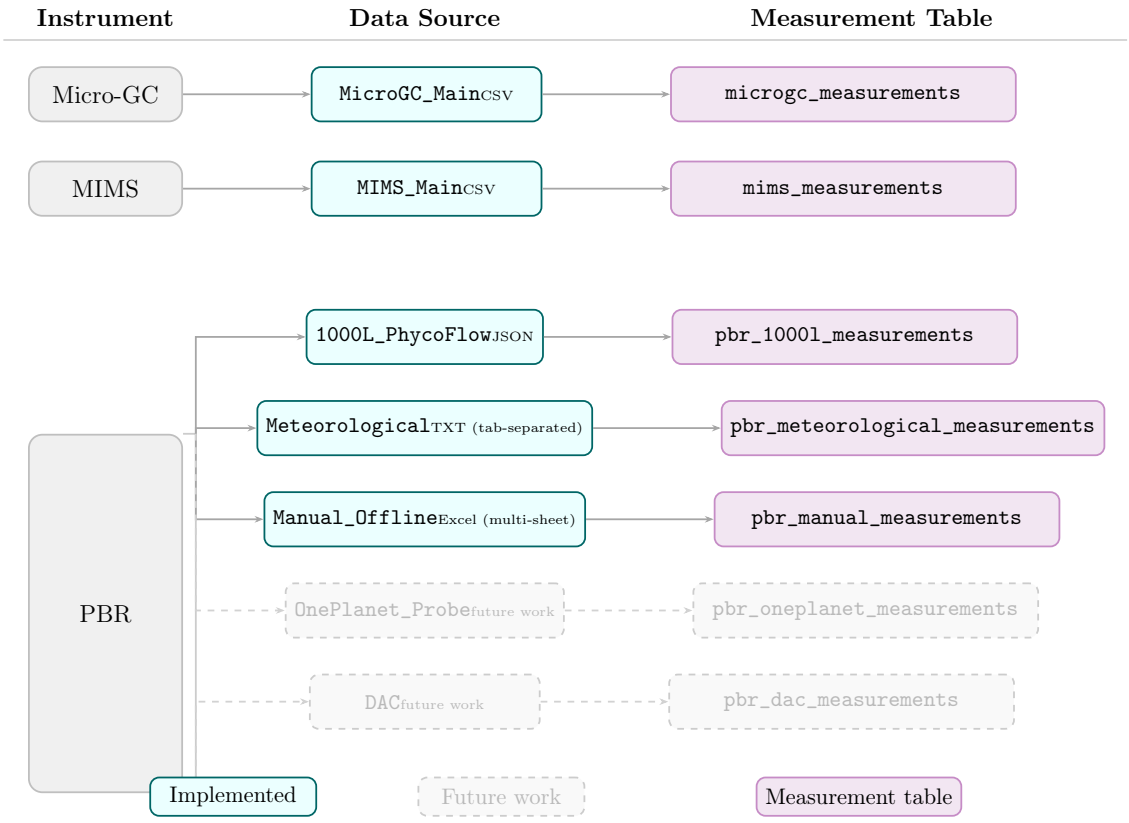

Figure 5.2: Mapping of instruments to data sources and measurement tables. The three instruments (Micro-GC, MIMS, PBR) produce five currently implemented data sources, each targeting a dedicated measurement table in the `photomicrobes_db` schema. The PBR instrument additionally has two planned data sources (OnePlanet Probe and DAC) identified as future work.

The `experiments` table constitutes the central organizing entity of the data model. In the current implementation, each experiment record links a named experimental run to a single instrument and data source through foreign key references and records

the start and end dates of the experimental period. It should be noted that this design reflects an instrument-dataset scope rather than a full biological experiment scope: in practice, a single biological experiment may simultaneously generate data from multiple instruments (for example, PBR monitoring combined with Micro-GC headspace analysis and short-term MIMS measurements). In such cases, the corresponding datasets would be stored as separate experiment records linked to their respective instruments and data sources. This is a known limitation of the current data model, and handling the association between co-occurring instrument datasets from a single biological experiment is identified as a direction for future development. Free-form contextual information, including organism identity, cultivation mode, and experimental conditions, is stored in an `experiment_metadata` JSONB column, which allows flexible, schema-on-write metadata capture without sacrificing the ability to query specific attributes when required [3].

5.3.2 Measurement Tables and the EAV Pattern

The system uses seven dedicated measurement tables, one per data source:

`microgc_measurements`, `mims_measurements`, `pbr_1000l_measurements`, `pbr_meteorological_measurements`, `pbr_manual_measurements`, `pbr_oneplanet_measurements`, and `pbr_dac_measurements`. Each table is a distinct relational entity implemented as a SQLAlchemy model that inherits from a shared base class defining the common column structure. This design preserves the ability to query each data source independently while avoiding schema duplication.

All seven measurement tables share a common Entity–Attribute–Value (EAV) column structure. Each row represents a single measurement of a single variable at a specific point in time, with the following columns:

- `id` — auto-incremented primary key
- `experiment_id` — foreign key referencing the parent experiment

- `timestamp` — the UTC datetime at which the measurement was recorded
- `variable_name` — the name of the measured variable (e.g., “Carbon dioxide”, “pH”, “temperature”)
- `value` — the numerical measurement value
- `unit` — the unit of measurement (e.g., “Area”, “°C”, “g/L”, “%”)
- `raw_data` — a JSONB column storing the original unparsed data record, preserving full provenance
- `created_at` — the timestamp at which the record was inserted into the database

In terms of the Entity–Attribute–Value model, the entity in this schema is the measurement event, identified by the combination of `experiment_id` and `timestamp`; the attribute is the `variable_name`; and the value is the numeric `value`, qualified by its `unit`. Rather than dedicating a single column to the entity, the schema identifies it compositely through the experiment reference and the measurement timestamp, while the remaining columns (`id`, `raw_data`, and `created_at`) serve as a surrogate key, provenance store, and insertion audit field, respectively. This long-format representation is what allows a single table structure to accommodate an arbitrary number of measured variables per instrument without schema modification.

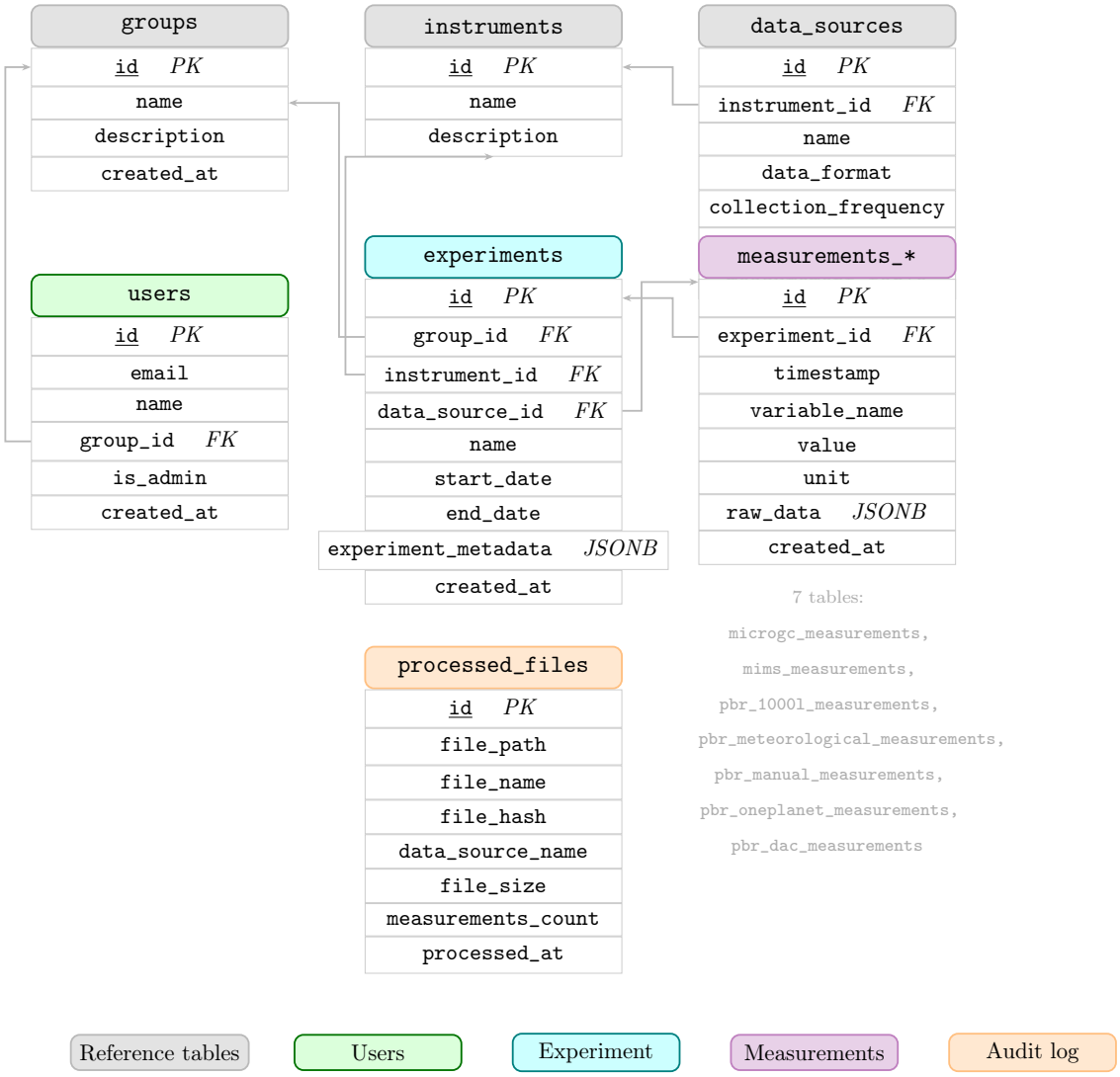

Figure 5.3: Detailed database schema showing table-level attributes and relationships. Reference tables (`groups`, `instruments`, `data_sources`) form the organizational hierarchy. The `experiments` table links all reference entities and stores contextual information in a JSONB column. Seven measurement tables share a common EAV structure with a `raw_data` JSONB provenance field. The `processed_files` table supports MD5-based deduplication and auditability. The `users` table stores researcher accounts with group membership and administrative role flags.

The EAV pattern was selected because it accommodates high variability in the number and names of measured variables across instruments without requiring schema alterations when new variables are introduced [36]. The inclusion of the `raw_data` JSONB column alongside the parsed `value` field is a deliberate provenance mechanism, ensuring that the original instrument output is preserved alongside the

normalized representation [40].

5.3.3 Provenance Tracking: the `processed_files` Table

The `processed_files` table serves as an audit log of all files that have been processed by the ingestion pipeline. Each record captures the full file path and file name of the source file, the name of the data source, the file size in bytes, the timestamp at which processing occurred, the number of measurement rows successfully inserted, and an MD5 file hash used for de-duplication and integrity verification.

This table fulfills several important functions. First, it enables the ingestion pipeline to detect previously processed files by comparing the hash of a candidate file against stored hashes, preventing duplicate ingestion [39]. Second, it provides a traceable record of the data provenance chain. Third, the `measurements_count` field allows administrators to verify that the expected volume of data was extracted from each processed file.

5.3.4 Access Control: the `users` Table

The `users` table manages researcher-level access to the system. Each user record captures an institutional email address, a display name, a foreign key reference to the associated research group, an `is_admin` boolean flag distinguishing administrative from standard users, and a `created_at` timestamp.

5.3.5 Data Flow

The end-to-end data flow of the system proceeds through four stages.

Stage 1 — Source file generation: Instruments generate output files in their native formats at the frequencies described in Section 4.3 (Table 4.1).

Stage 2 — Ingestion pipeline execution: Ingestion scripts detect new files in designated source network directories, validate their format and structure, parse

measurement values into normalized rows, and load them into the appropriate measurement table in the central database. The `processed_files` table is updated atomically with each successful ingestion of a data file.

Stage 3 — Structured storage: Parsed measurements are stored relationally, linked to their parent experiment through the experiment identifier foreign key. Contextual metadata is preserved in the database alongside its respective experiments.

Stage 4 — Access and visualization: Researchers access, view, and interact with stored data through a web interface that provides dataset browsing, file upload, and interactive time-series chart plotting.

5.4 Data Ingestion and Storage

The data ingestion layer constitutes the operational core of the proposed system, responsible for bridging the gap between heterogeneous instrument outputs and the structured relational storage layer. It implements an ETL (Extract, Transform, Load) pipeline adapted to the heterogeneous file formats produced by the research instruments: raw files are extracted from monitored source directories, transformed through format-specific parsing and controlled vocabulary normalization into structured EAV rows, and loaded into the central database. Instrument-specific ingestion scripts have been developed in Python for all three primary instruments, the Micro-GC, MIMS, and PBR, each implemented and tested against representative experimental datasets. Each script follows a common processing pipeline: file detection, format validation, data parsing, value normalization, and database insertion, while accommodating the format-specific characteristics of its target instrument.

File detection is driven by an event-based file system watcher implemented using the `watchdog` library. A `DataFileHandler` monitors designated source directories on instrument-connected computers and triggers ingestion automatically when a new file matching the expected format is created. A `multi_watcher` orchestrator

manages watchers across all configured data sources simultaneously. This event-driven approach ensures that new instrument data is ingested with minimal latency as soon as it becomes available in the monitored directory, without requiring manual intervention or a fixed polling schedule.

5.4.1 Ingestion Pipeline Architecture

The ingestion pipeline is structured as a sequence of discrete processing stages, following established patterns for modular data pipeline construction [36]. This separation of concerns allows each stage to be tested and modified independently, reducing the risk of cascading errors during format changes or schema updates [41].

Stage 1 — File detection and de-duplication: The file system watcher triggers pipeline execution when a new file matching the expected format appears in a monitored source directory. Before proceeding with parsing, the script computes an MD5 hash of the candidate file and queries the `processed_files` table to determine whether a record with the same hash already exists. If a match is found, the file is skipped and the event is logged. This hash-based de-duplication mechanism prevents redundant data ingestion in cases where files are re-copied or the watcher is restarted over an existing directory [39].

Stage 2 — Format validation: After confirming that a file has not been previously processed, the script performs structural validation to confirm that the file conforms to the expected format. For CSV-based sources (Micro-GC and MIMS), this includes verifying the presence of required column headers and confirming that the file is non-empty and well-formed. Files that fail validation are logged with a descriptive error message and excluded from ingestion without interrupting the processing of subsequent files.

Stage 3 — Parsing and normalization: Validated files are read into memory using the `pandas` library [38]. Each row of the source file is transformed into

a normalized EAV record comprising a `timestamp`, a `variable_name`, a numeric `value`, and a `unit`. Variable names and units are mapped to a controlled vocabulary defined within the script, ensuring that labels remain consistent across ingestion runs. The original unparsed row is simultaneously serialized as a JSON object and stored in the `raw_data` JSONB column of the measurement table, preserving full provenance alongside the normalized output [40].

Stage 4 — Database insertion: Normalized measurement rows are inserted in bulk into the appropriate measurement table in the central database using `SQLAlchemy` [37]. Each insertion is wrapped in a database transaction, ensuring that partial failures do not result in incomplete records. On successful completion, a corresponding record is written to the `processed_files` table, capturing the source file path, MD5 hash, data source name, file size, ingestion timestamp, and the number of measurement rows successfully inserted.

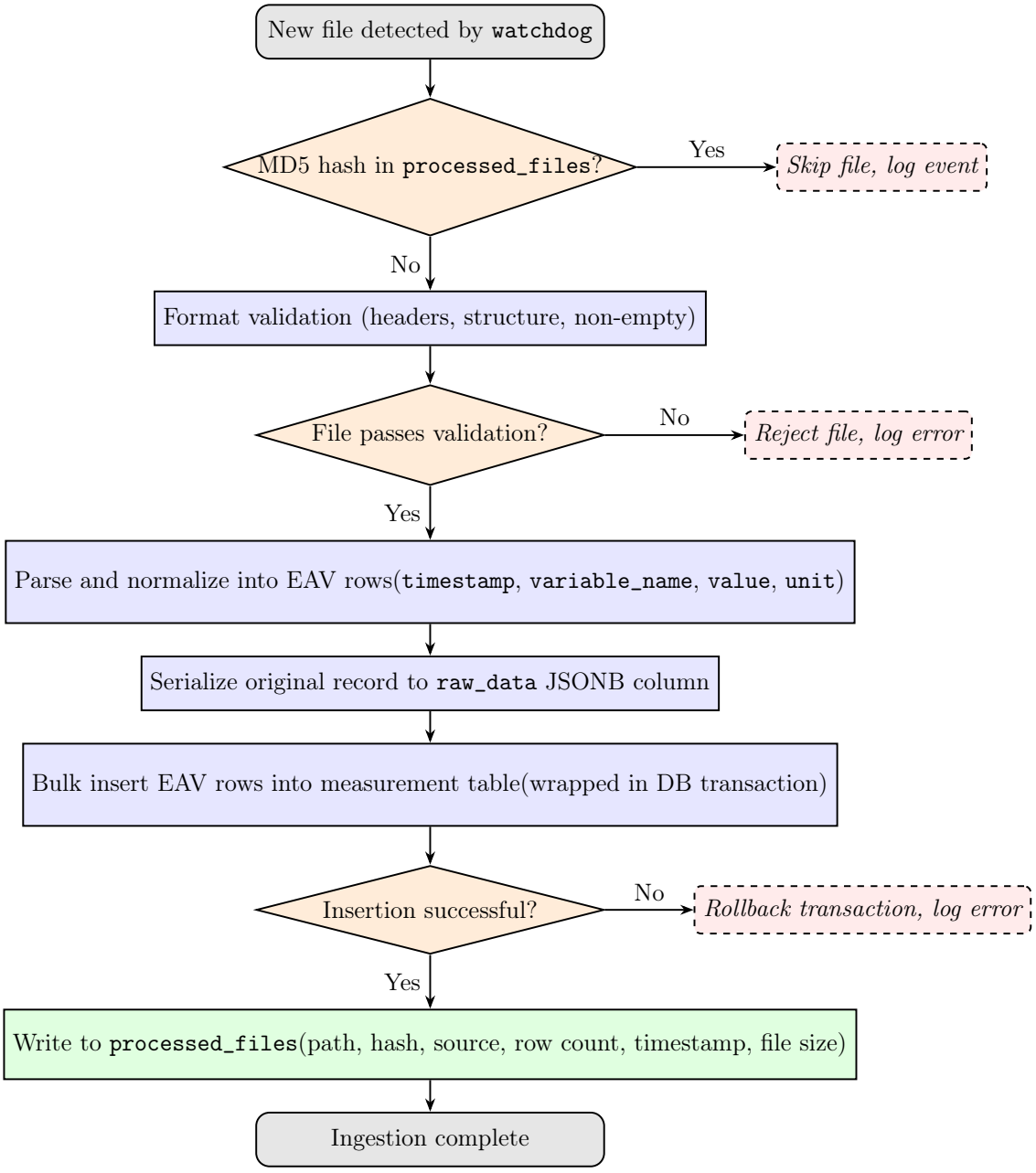

Figure 5.4: Ingestion pipeline flowchart. Each new file detected by the `watchdog` file system watcher passes through four stages: hash-based de-duplication, format validation, parsing and normalization into EAV rows, and transactional database insertion. Failed or duplicate files are logged and skipped without interrupting subsequent files. Successful ingestion events are recorded in the `processed_files` audit table.

5.4.2 Instrument-Specific Parsing Logic

Each instrument produces data in a format specific to its hardware and software environment, requiring a dedicated parsing script that understands the file structure, extracts the relevant measurement values, maps variable names to the controlled vocabulary, and transforms the raw records into normalized EAV rows. The three implemented parsers, for the Micro-GC, MIMS, and PBR, follow the same four-stage pipeline but differ in their file format handling, timestamp extraction logic, and variable mapping. The subsections below describe each parser in turn.

Micro-GC

The instrument exports summarized results as CSV files with a fixed column structure containing retention times, peak areas, and calibrated gas concentrations for each detected species. The Micro-GC is capable of quantifying gases including H_2, O_2, N_2, and CO_2; the current ingestion configuration targets Carbon dioxide, Ethylene, Oxygen, and Nitrogen as the active measurement variables. The parser reads the file using `pandas`, extracts the `Acquired_Date` column present in the CSV data rows as the measurement timestamp, and transforms each target gas column into a separate EAV row. Each run is treated as a single logical batch, and all EAV rows derived from it share the same `experiment_id` foreign key.

Testing confirmed that the script correctly handles multi-run CSV exports, detects and skips previously ingested files via the MD5 hash check, and rejects malformed files at the validation stage without aborting the overall pipeline execution.

MIMS

The MIMS system produces time-series CSV files in which each row corresponds to a timestamped measurement of one or more ion-current channels. The parser reads the file sequentially and maps each channel to its corresponding gas species and

isotopologue label using the controlled vocabulary mapping, generating one EAV row per channel per timestep. In the group's current experimental workflows, individual MIMS sessions typically last between 20 and 40 minutes. The parser processes files using `pandas` chunked reading to manage memory consumption efficiently, particularly for files that may contain a large number of timestep rows [38]. It should also be noted that MIMS experiments may include offline data recorded alongside the instrument measurements, similar to PBR experiments, and such data would be captured through the Manual Offline data source.

Testing confirmed that the script correctly handles the structured CSV output, preserves timestamp ordering in the inserted records, and accurately maps ion-current channel labels to standardized variable names.

PBR

The PBR instrument produces data from five distinct data sources. Of these, three are currently implemented: the 1000L PhycoFlow (JSON), Meteorological (tab-separated TXT), and Manual Offline (Excel). The OnePlanet Probe and DAC data sources are identified as future implementation work.

PBR parsing is handled by dedicated parser classes in `pbr_parsers.py`. The `PBR1000LParser` handles the primary PhycoFlow JSON data stream: the 1000L PBR logs operational data as JSON files at approximately 30-second intervals, each capturing a snapshot of up to 25 environmental and operational variables including temperature, pH, dissolved oxygen, gas flow rates, and light intensity. The parser reads each JSON file, iterates over its data array, extracts numeric variable-value pairs, and generates one EAV row per variable per timestep [42]. Because the PBR produces a high volume of small files, the `multi_watcher` orchestrator processes all JSON files in a monitored directory sequentially, invoking the parser once per file, with the MD5-based de-duplication check applied per file to prevent redundant

insertion of previously ingested records. The `PBRMeteorologicalParser` handles Finnish-language tab-separated TXT files from the greenhouse monitoring system, and the `PBRManualParser` handles Excel-based offline measurement records.

Testing confirmed that the `PBR1000LParser` correctly handles directories containing large numbers of JSON snapshots, accurately filters non-numeric fields, and maps PBR-native variable keys to standardized variable names and units using the controlled vocabulary mapping.

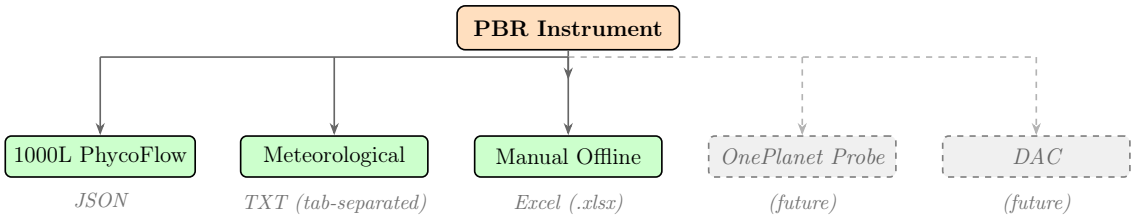

Figure 5.5: Data source tree for the PBR instrument. The PBR produces data from five distinct sources. Three are currently implemented: the 1000L PhycoFlow (JSON), Meteorological (Finnish-language tab-separated TXT), and Manual Offline (Excel). The OnePlanet Probe and DAC data sources are identified as future implementation work.

5.4.3 Execution and Scheduling

The primary execution model for the ingestion pipeline is event-driven: the `watchdog`-based file system watcher continuously monitors source directories and triggers ingestion immediately when a new file is detected. This approach is well-suited to the PBR's high-frequency output, where new JSON files arrive every 30 seconds, and to the Micro-GC and MIMS, where files are produced at irregular intervals. Where continuous monitoring is not practical, the ingestion scripts can alternatively be invoked on a scheduled basis as a complementary execution mode.

5.5 Data Models and Metadata Structure

Effective data management in experimental research depends not only on the physical organization of stored measurements but also on the richness of the metadata that contextualizes them. Without sufficient metadata, datasets risk becoming uninterpretable over time, undermining both reproducibility and reuse [10]. The metadata structure of the proposed DMS has been designed with this in mind, balancing the need for structured, queryable attributes with the flexibility required to accommodate the heterogeneous experimental contexts of the Photosynthetic Microbes group.

5.5.1 Hierarchical Data Model

The data architecture of the DMS follows a hierarchical model in which both measurement data and contextual metadata are organized at multiple levels of granularity, progressing from the organizational unit down to the individual measurement record. This layered approach reflects established principles in research data repository design, which emphasize that contextual information should describe both the content and the provenance of data at each level of abstraction [39].

At the top of the hierarchy, the `groups` table captures organizational-level metadata, identifying the research group responsible for all experimental activity within the system. The `instruments` table records instrument-level metadata, including the instrument name, type, and a free-text description. The `data_sources` table extends this with source-level metadata, linking each data stream to its parent instrument and recording the expected file format and ingestion target table.

Experiment-level metadata is the most semantically rich layer of the hierarchy and is captured in two complementary forms within the `experiments` table. Structured attributes, including the experiment name, associated group, instrument, data source, and the start and end dates of the experimental period, are stored as typed relational

columns, supporting deterministic querying and foreign key integrity. Free-form contextual metadata is stored in the `experiment_metadata` JSONB column. For photosynthetic experiments, the following fields are treated as mandatory and must be provided at the time of experiment registration: organism or strain identifier, growth medium, cultivation mode (batch, semi-continuous, or continuous), light conditions, temperature, gas supply, instrument settings, calibration information, operator name, and the experimental protocol applied. Additional free-form attributes may be recorded alongside these required fields. This design allows metadata to be extended without schema migrations, while retaining the ability to query specific attributes using PostgreSQL's native JSONB operators [3].

At the measurement level, provenance metadata is embedded directly in each EAV row through the `raw_data` JSONB column, which preserves the original unparsed instrument record alongside the normalized value. This ensures that the full context of each measurement is retained even after normalization, supporting retrospective verification and reprocessing [40].

5.5.2 Controlled Vocabulary for Variable Names and Units

A persistent challenge in multi-instrument research data management is the inconsistency of variable naming conventions across different instruments and operators [18]. To address this, the ingestion scripts enforce a controlled vocabulary for the `variable_name` and `unit` fields of all EAV records. Each script contains an internal mapping dictionary that translates instrument-native column headers or JSON keys into standardized labels before insertion. Table 5.1 provides representative examples of these mappings across all three instruments, including variables where the same physical quantity is measured by different instruments.

It is important to note that the same physical parameter may be measured by different instruments. For example, CO_2 and O_2 can be monitored simultaneously

Table 5.1: Representative controlled vocabulary mappings for instrument-native labels to standardized `variable_name` and `unit` values.

Instrument	**Native label**	**variable_name**	**unit**
Micro-GC	`Ethylene [area]`	Ethylene	Area
Micro-GC	`CO2 [area]`	Carbon dioxide	Area
Micro-GC	`O2 [area]`	Oxygen	Area
Micro-GC	`H2 [area]`	Hydrogen	Area
MIMS	`m/z 32`	Oxygen-32	nA
MIMS	`m/z 44`	Carbon dioxide-44	nA
MIMS	`m/z 2`	Hydrogen-2	nA
MIMS	`m/z 34`	Oxygen-34	nA
PBR	`temp_culture`	Culture Temperature	°C
PBR	`do_percent`	Dissolved Oxygen	%
PBR	`co2_flow`	CO2 Flow Rate	L/min
PBR	`o2_dissolved`	Dissolved Oxygen Sensor	mg/L

by the Micro-GC (headspace gas composition), MIMS (gas exchange in liquid), and PBR sensors (dissolved gases and flow rates). The controlled vocabulary assigns instrument-specific standardized names to each of these, ensuring that queries correctly distinguish between measurements of the same parameter from different sources. This cross-instrument mapping is an important consideration for downstream data analysis, particularly as the group plans experiments in which PBR monitoring, Micro-GC headspace analysis, and MIMS measurements are conducted simultaneously on the same biological experiment. Handling the association of these co-occurring datasets at the query level is identified as a direction for future development; in the current implementation, it is recommended that experiment names and metadata fields be used consistently to group related instrument datasets from the same biological experiment.

This normalization ensures that queries against `variable_name` and `unit` yield consistent results regardless of which instrument or operator generated the data. The

controlled vocabulary is maintained as a Python dictionary within each ingestion script and is version-controlled alongside the script code, providing a transparent and auditable record of all label mappings [1].

5.5.3 Data Completeness and FAIR Alignment

The system has been designed with reference to the FAIR data principles [10]. Findability is supported through structured experiment records that can be queried by instrument type, organism, date range, and researcher. Accessibility is provided through the web dashboard interface, which exposes these attributes as search and filter parameters without requiring direct database credentials. Interoperability is addressed through the use of standardized variable names and units, which reduce the semantic distance between datasets produced by different instruments. Reusability is supported through the `raw_data` provenance field and the `experiment_metadata` JSONB column, which together preserve sufficient context for results to be correctly interpreted and reproduced by other researchers [14].

Full FAIR compliance, including the assignment of persistent identifiers such as DOIs to individual datasets and the publication of data in a machine-readable catalogue, is beyond the current scope of the system. However, the system architecture has been designed to be extensible in this direction, and the structured `experiments` table provides a natural foundation for future integration with institutional data catalogues or open repositories [10].

5.6 Access, Querying, and Visualization

The access and visualization layer provides researchers with a web interface for data browsing, file upload, and interactive visualization, backed by a FastAPI REST API that exposes structured endpoints for experiments, measurements, data sources,

upload, and export. This section describes how users access the system, how data is queried, and how measurement data is visualized.

5.6.1 User Access and Permissions

Researchers access the system through a web browser using credentials registered in the `users` table. Upon login, users are presented with the web application interface described in Section 5.6.3. Standard users can browse experiments, view and filter measurement data, upload instrument files for ingestion, and export data in CSV format. Administrative users additionally have access to experiment registration and user management functions. All access is governed by the role-based security model described in Section 5.7.

5.6.2 Backend: REST API

The system backend is implemented as a FastAPI application exposing a REST API with structured endpoints for experiments (`/api/experiments`), measurements (`/api/measurements`), data sources (`/api/data-sources`), file upload (`/api/upload`), and data export (`/api/export`). This API constitutes the sole communication layer between the frontend and the database, and also enables external access to all system data via the API independently of the web interface.

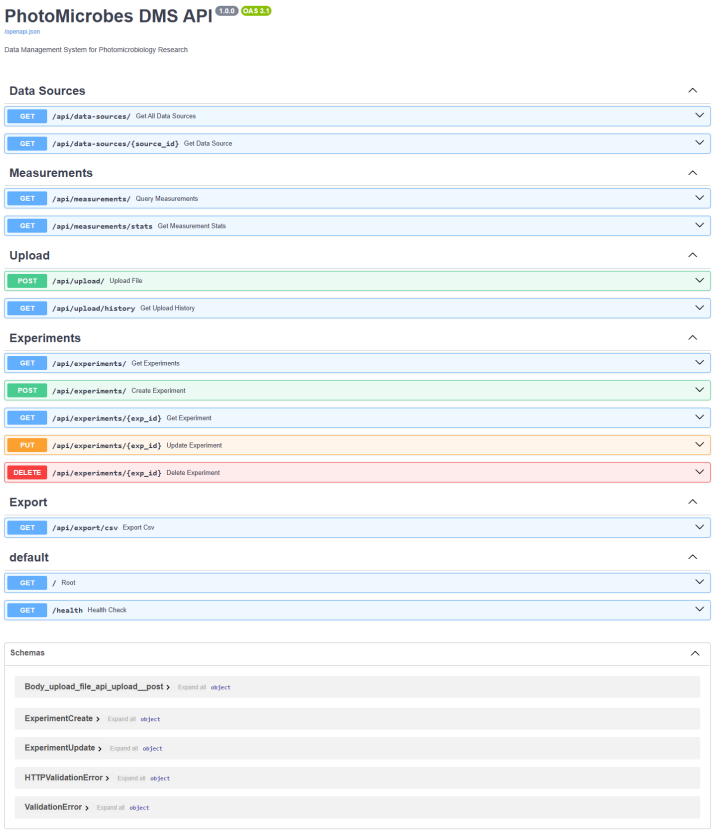

Figure 5.6: FastAPI auto-generated Swagger UI showing the full set of REST API endpoints exposed by the system backend, including routes for experiments, measurements, data sources, file upload, and data export.

5.6.3 Frontend: Web Application

The frontend is a React single-page application (SPA) built with Vite, served separately from the backend. It currently provides the following views:

- **Homepage**: A summary view displaying system-level statistics, including the number of registered experiments, data sources, and total ingested measurement rows. A screenshot of this view is shown in Figure 5.7.
- **Data Sources**: A view listing all registered data sources with their associated instrument, format, and ingestion target table. Figure 5.8 shows the Data Sources page with the registered instruments and their configurations.

- **Upload**: A web-based file upload interface allowing researchers to submit instrument data files directly through the browser. The interface presents a file picker for selecting the instrument data file, a dropdown menu for selecting the target data source (e.g. MicroGC_Main, Meteorological, Manual_Offline), and an Upload & Process button that triggers ingestion. Uploaded files are processed by the ingestion pipeline via the `/api/upload` endpoint, applying the same deduplication and validation logic as watcher-triggered ingestion. On successful processing, the interface displays a confirmation message showing the number of measurements stored. This provides a manual submission pathway complementary to the automated file watcher, particularly useful for instruments whose data is collected manually, such as the Meteorological and Manual Offline sources. Figure 5.9 shows the Upload page interface.

- **Dashboards**: A visualization view rendering measurement charts for data directly coming from the database, enabling exploratory data analysis directly within the web interface. A representative visualization is shown in Figure 5.10.

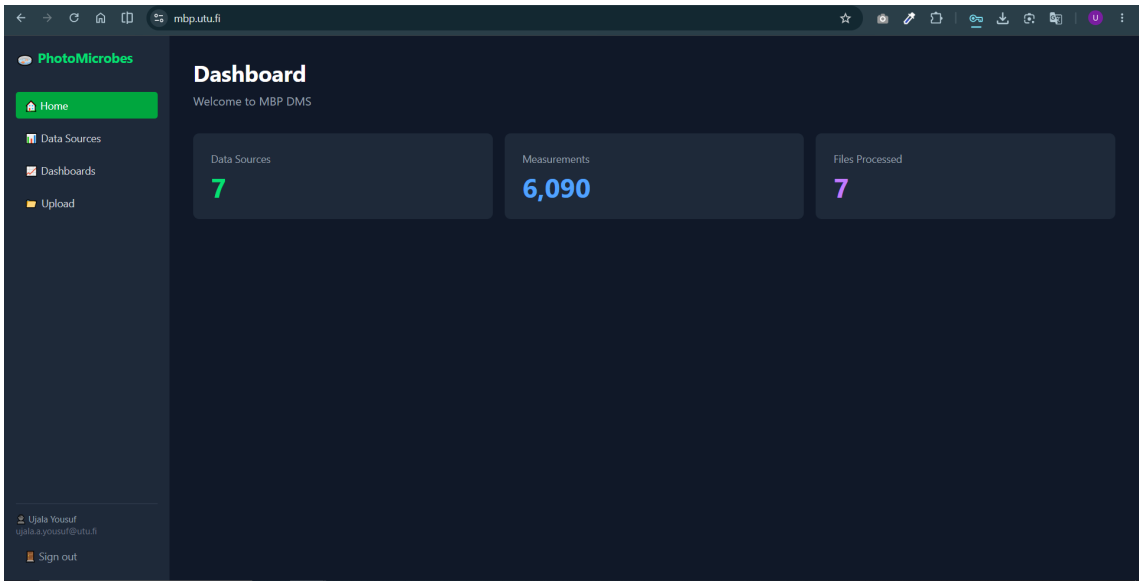

Figure 5.7: Home view of the web application, displaying the number of data sources, measurement entries, and ingested measurement rows.

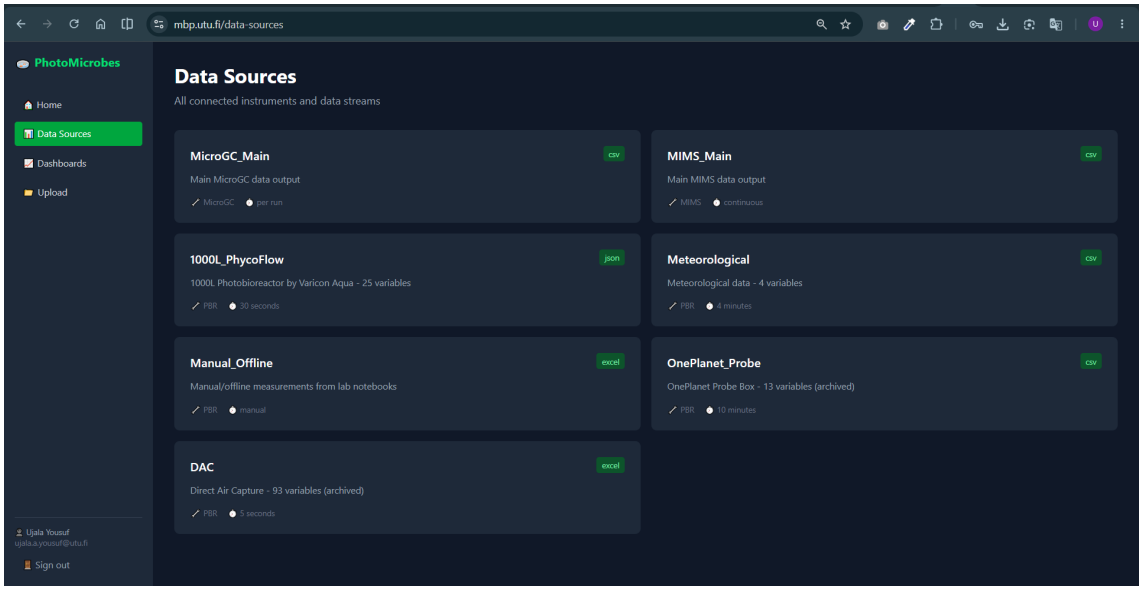

Figure 5.8: Data Sources view listing registered instruments, data source types, file formats, and ingestion target tables.

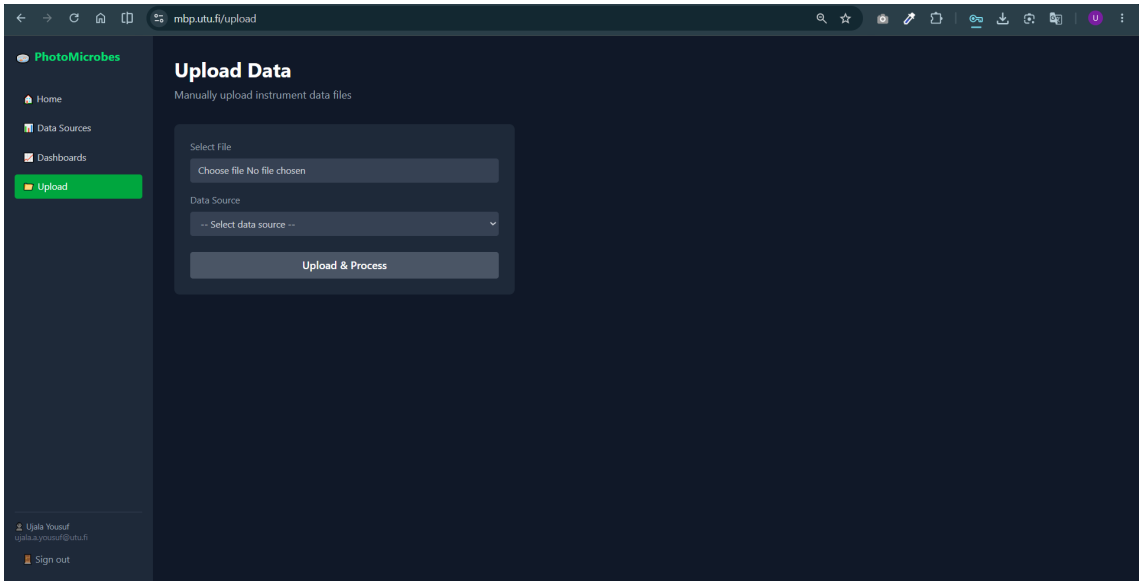

Figure 5.9: Upload view providing a browser-based file submission interface for manual data ingestion.

Access to the application is governed through the application-level authentication mechanism described in Section 5.7. The interface design prioritizes simplicity and accessibility, consistent with the finding from the literature review that researcher-centric usability is a critical determinant of RDM system adoption [25].

5.6.4 Web-Based Visualization Interface

The web application provides an integrated visualization interface through which researchers can explore ingested measurement data directly in the browser, without requiring a separate visualization tool or database credentials. The frontend is built using React, an open-source frontend library, and the data charts are plotted using Recharts, a composable charting library for React built on top of D3.js [43], which was selected because it integrates natively into the React SPA and allows the visualization and data browsing functionality to be delivered as a single unified application.

The Dashboards view renders charts for ingested measurement data, populated by querying the `/api/measurements` endpoint, which returns records filtered by experiment, instrument, and variable. Researchers select the experiment and variable of interest from interactive variable assignment controls within the interface, and the corresponding chart updates accordingly. The chart type and interpretation vary by instrument:

- **Micro-GC**: Measurement run charts display calibrated gas concentrations (Carbon dioxide, Ethylene, Oxygen, Nitrogen) across discrete measurement runs within a selected experiment. Because the Micro-GC produces one result per run rather than a continuous stream, these charts show run-by-run values rather than continuous time series.

- **MIMS**: Time-series charts display ion-current readings per mass channel across the duration of a measurement session, reflecting the structured time-series output of the MIMS instrument.

- **PBR**: Multi-variable time-series charts display environmental and operational parameters, temperature, pH, dissolved oxygen, gas flow rates, and light intensity, over the full duration of a cultivation experiment, reflecting the

30-second sampling interval of the PhycoFlow system.

Figure 5.10 shows the Dashboards view with a representative visualization rendered from ingested instrument data.

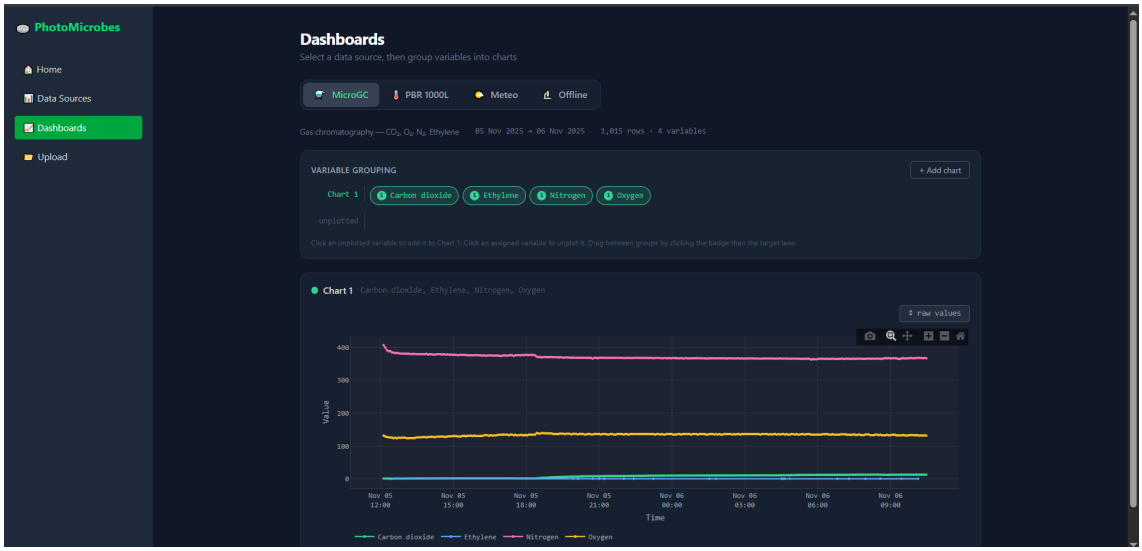

Figure 5.10: Dashboards view showing Micro-GC gas concentration measurements.

Figure 5.11: Dashboards view showing PBR 1000L PhycoFlow time-series data.

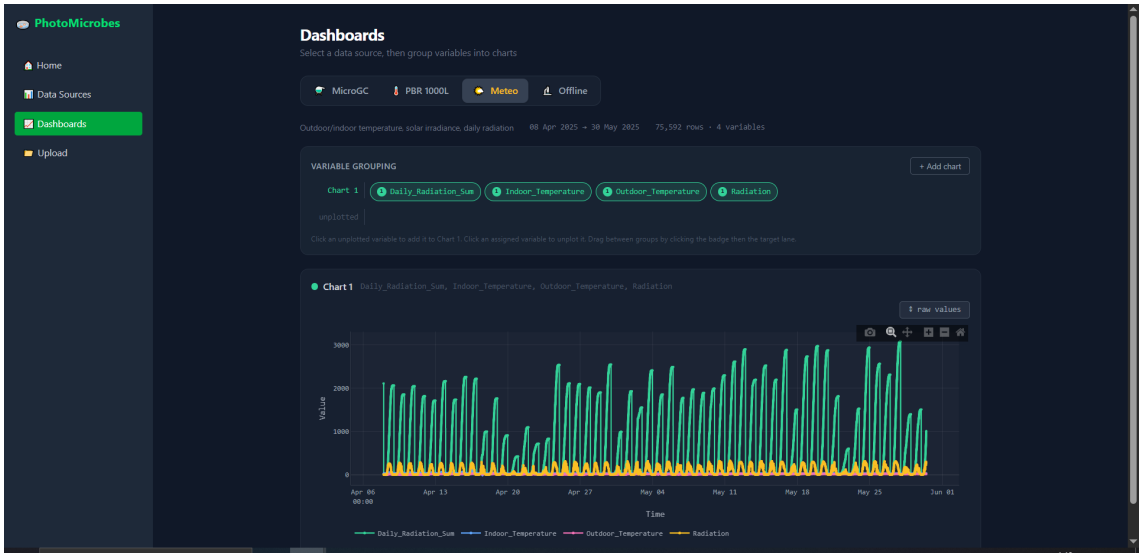

Figure 5.12: Dashboards view showing Meteorological data measurements.

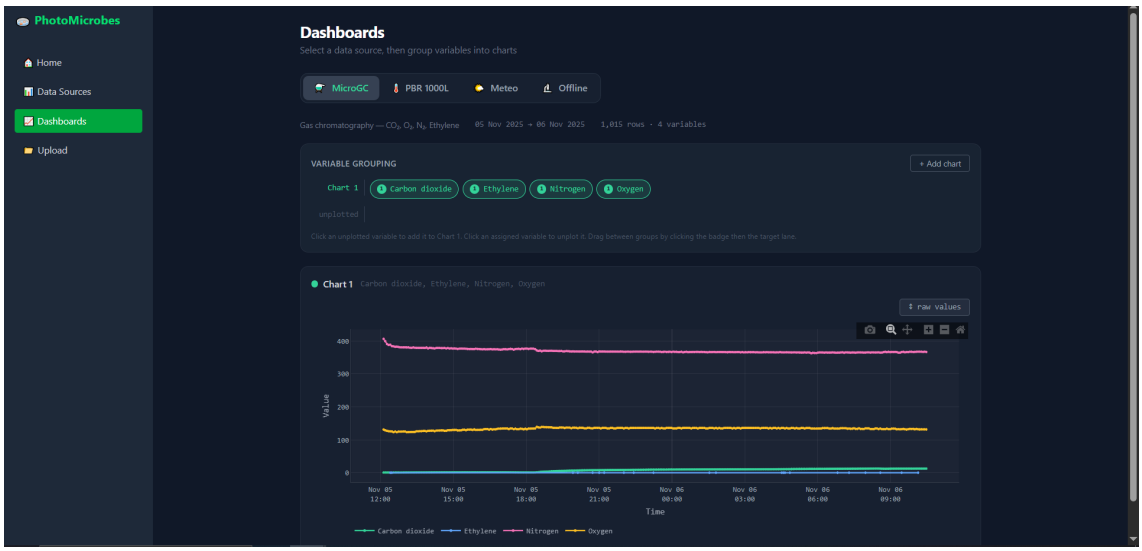

Figure 5.13: Dashboards view showing Manual Offline measurement data.

5.6.5 Querying Stored Data

The PostgreSQL schema supports a range of analytical queries against stored data directly via SQL. During development and testing, DBeaver was used as the database client to connect directly to the CSC Pukki PostgreSQL instance, enabling schema inspection, query execution, and data verification. The EAV structure of the measurement tables supports cross-experiment queries, such as retrieving all measurements of

a given variable across all experiments within a specified date range. This is achieved through a standard `WHERE` clause on `variable_name` combined with a join to the `experiments` table on `experiment_id`. The hierarchical foreign key structure, from `groups` through `instruments` and `data_sources` to `experiments` and measurement tables, enables queries to be scoped at any level of the data model [3].

5.7 Security and Access Control Considerations

The security architecture of the system is designed to protect unpublished experimental data from unauthorized access while remaining simple enough to be maintained by a small development team. This balance reflects the requirements articulated in Section 4.4 (NFR3 and NFR5) and the practical constraints of deploying within an academic research environment [1].

5.7.1 Database-Level Access Control

Access to the PostgreSQL database instance on CSC Pukki (`Photosynthetic Microbes_db`) is governed by the platform's built-in user account management [37]. Three database roles are defined to enforce the principle of least privilege:

- **Admin role**: Holds full read and write privileges across all tables. Used exclusively by the ingestion pipeline scripts and by the system administrator for schema maintenance operations.
- **Application role**: Holds read-only privileges on all measurement and metadata tables. Used by the web application to query stored data without the ability to modify it.
- **Researcher role**: An optional read-only role that is granted to all the members of a research group to view and access their data on the data management

system. Assignment of this role is managed by the system administrator.

Database credentials are not stored in version-controlled code. Instead, they are passed to the ingestion scripts and the web application via environment variables, a standard practice for credential management in Python-based applications that prevents accidental exposure through code repositories [36].

5.7.2 Application-Level Access Control

The web dashboard implements a session-based authentication layer using username and password credentials stored in the `users` table of the `Photosynthetic Microbes_db` schema. Passwords are stored as hashed values rather than plaintext. The `is_admin` boolean flag in the `users` table governs access to administrative functions within the dashboard, such as registering new experiments or managing user accounts, while standard users are restricted to read and export operations.

Role-based access control of this kind is well established as a practical mechanism for protecting research data within collaborative groups, offering meaningful protection against inadvertent or unauthorized modification without imposing the administrative overhead of more complex identity management systems [1].

5.7.3 Network and Transport Security

All connections between the web application and the CSC Pukki PostgreSQL endpoint are established over TLS-encrypted connections, consistent with the security requirements of the CSC platform [37]. The web application is deployed on CSC Pouta and served over HTTPS, limiting its external exposure to authorized users within the University of Turku network. Data in transit between the instrument-connected computers and the database is encrypted at the transport layer as part of the standard CSC Pukki connectivity model.

5.7.4 Data Retention and Backup

CSC Pukki provides automated daily backups of database instances as part of its managed service offering [37], providing a baseline level of protection against data loss without requiring manual backup procedures. Raw instrument files are retained on the instrument-connected computers and are not deleted after ingestion, ensuring that the original source data remains available for reprocessing in the event of pipeline errors or schema changes. This dual-retention approach - parsed measurements in the database alongside raw files on local storage- supports both operational continuity and long-term data provenance [14].

5.8 Integration and Interoperability Considerations

The system has been designed to work effectively within its own operational boundaries first, while making architectural choices that keep the path toward broader integration open. Rather than attempting to conform to every external data exchange standard from the outset, which would add significant complexity to an early-stage system, the design focuses on internal consistency, open formats, and a clean API layer. These choices lower the cost of future integration with institutional repositories, federated research platforms, and external analysis tools, without making such integration a prerequisite for the system to deliver value to the research group.

5.8.1 Integration with Existing Research Infrastructure

The primary integration point of the DMS is with the CSC research computing infrastructure. The choice of CSC Pukki as the database hosting platform ensures that the system is deployed within institutional infrastructure that is already trusted, maintained, and accessible to University of Turku researchers [37]. This eliminates the need for the Photosynthetic Microbes group to manage its own database server

and aligns the system with the University of Turku's broader data management strategy.

The CSC Allas object storage service [44] has been identified as a complementary infrastructure component for the long-term archival of raw instrument files. While raw files are currently retained on instrument-connected local storage, a planned integration pathway involves periodic upload of raw files to Allas buckets, providing geographically redundant, institutionally managed cold storage for original instrument outputs. This would extend the provenance chain from the database's `raw_data` JSONB fields back to the original binary source files, further strengthening long-term data accessibility [14].

5.8.2 Data Format Interoperability

Within the system, interoperability between the ingestion layer and the storage layer is achieved through the consistent use of open, well-documented formats. CSV serves as the normalized exchange format between instrument export software and the ingestion scripts. JSON is used for PBR data and for the `raw_data` and `experiment_metadata` JSONB columns within PostgreSQL, following the widely adopted JSON data interchange standard [42]. The use of open formats at both the ingestion and storage layers ensures that data can be accessed and reprocessed using standard tools without dependency on proprietary software.

The EAV measurement schema is format-agnostic by design: any instrument whose output can be reduced to timestamped variable-value pairs can be integrated into the system by developing a new ingestion script with an appropriate parsing and controlled vocabulary configuration. This extensibility directly addresses one of the core limitations of existing RDM systems identified in the literature review — namely, the difficulty of integrating new instrument types without system-level reconfiguration [21].

5.8.3 Alignment with FAIR Principles

The current implementation partially addresses the FAIR data principles [10] within its operational scope. Findability is supported through structured experiment metadata that can be queried via the web application. Accessibility is provided through the web interface, which exposes stored data without requiring direct database credentials. Interoperability is addressed through standardized variable names, units, and open file formats. Reusability is supported through the combination of provenance metadata, controlled vocabulary enforcement, and CSV export functionality.

Full FAIR compliance, particularly the assignment of globally persistent identifiers to datasets and the publication of machine-readable records in a public catalogue, remains a future development goal. The current system lays the architectural groundwork for this: the structured `experiments` table is a natural source for dataset-level metadata records, and the open formats used throughout the ingestion and storage layers are compatible with standard repository submission workflows [9].

5.8.4 Limitations and Future Integration Pathways

The current system does not implement semantic interoperability in the form of ontology-based metadata or linked data representations. Variable names and units are standardized through a controlled vocabulary, but this vocabulary is not formally aligned with established scientific ontologies such as the Chemical Entities of Biological Interest (ChEBI) or the Units of Measurement Ontology (UO). Such alignment would improve cross-system data integration and is a natural direction for future development, should the Photosynthetic Microbes group seek to contribute data to federated research data platforms [16].

Similarly, the system provides a REST API via FastAPI for programmatic data access, exposing endpoints for experiments, measurements, data sources, file upload, and data export. This API enables external tools and analysis pipelines to

interact with stored data without requiring direct database connectivity, improving interoperability with notebook-based environments such as Jupyter and with future data portals.

6 Evaluation and Results

6.1 Evaluation Criteria

The evaluation of the proposed data management system (DMS) is grounded in the design science research (DSR) framework adopted in Chapter 4. Within DSR, artifact evaluation constitutes a core phase in which the constructed artifact is assessed against the requirements that motivated its design [30], [31].

The evaluation presented in this chapter, therefore, follows two complementary methods. First, a *requirements traceability* assessment, in which each functional and non-functional requirement defined in Section 4.4 serves as an explicit evaluation criterion. Second, a *comparative analysis* against existing research data management solutions identified in Chapter 3, examining how the proposed DMS addresses the gaps and limitations found in those systems.

6.1.1 Functional Requirements Criteria

The six functional requirements (FR1–FR6) correspond directly to the core capabilities the DMS must provide to address the problem identified in Section 1.2. Table 6.1 summarizes each requirement, its evaluation method, and the implementation status at the time of writing.

Table 6.1: Functional requirements and their evaluation criteria.

ID	Criterion	Evaluation Method
FR1	Ingestion of heterogeneous data from Micro-GC (CSV), MIMS (CSV), and PBR (JSON)	Demonstrate pipeline execution on real instrument files; verify row insertion and record count in the `microgc_measurements`, `mims_measurements`, and `pbr_1000l_measurements` tables.
FR2	Standardized naming conventions applied at ingestion	Inspect `variable_name` and `unit` field values across ingested rows; verify controlled vocabulary consistency across multiple ingestion runs from different instruments.
FR3	Centralized, cloud-accessible repository accessible to authorized project members through role-based authentication	Confirm the database instance is live and reachable via the connection credentials provided by CSC Pukki; verify that the role-based access model correctly restricts write access to the admin role while permitting read access through the application role.
FR4	Browsing and querying of datasets by experiment, instrument, date range, and researcher	Demonstrate filtering and retrieval in the web interface; verify that query results match expected records across multiple filter combinations.
FR5	Visualization interface capable of rendering measurement data from stored datasets	Demonstrate that charts are plotted correctly on the web interface for all three instruments (Micro-GC, MIMS, PBR), with data retrieved from the REST API and rendered interactively in the browser.
FR6	Traceability of data origin, collection date, and applied transformations	Inspect `processed_files` table entries after ingestion; verify that MD5 hash, source file path, timestamp, and row count are recorded for each processed file; confirm deduplication by re-running the pipeline on an already-processed file.

6.1.2 Non-Functional Requirements Criteria

The six non-functional requirements (NFR1–NFR6) address quality attributes of the system that are not directly observable through functional testing but are nonetheless essential to its long-term usability and institutional fitness. Table 6.2 summarizes each non-functional criterion and its evaluation approach.

Table 6.2: Non-functional requirements and their evaluation criteria.

ID	**Criterion**	**Evaluation Method**
NFR1	Deployable within the University of Turku's CSC infrastructure without external proprietary services	Confirm that all system components, the PostgreSQL database (hosted on CSC Pukki), the ingestion pipeline, the FastAPI backend, and the React frontend, operate entirely within CSC infrastructure or open-source tooling; verify that no external paid services are required for operation.
NFR2	Minimized manual effort through automated ingestion pipelines	Verify event-driven file watcher execution; qualitatively compare manual steps required in the previous workflow against those required after system deployment.
NFR3	Extensible to support additional instrument types and data sources without requiring changes to the core database schema	Design-level assessment: verify that the EAV schema and shared base model allow a new measurement table to be added without altering existing tables; confirm that the `data_sources` table can register a new source without schema modification; assess whether a new ingestion script can be added following the established template without changes to shared pipeline infrastructure.

ID	Criterion	Evaluation Method
NFR4	Compliance with FAIR data principles within operational scope [10]	Map each FAIR principle to specific system features: Findable (structured experiment records and controlled vocabulary), Accessible (database credentials and role-based access), Interoperable (standardized field naming and open file formats), Reusable (provenance records in `processed_files` and raw JSONB preservation). Identify remaining gaps.
NFR5	Role-based access controls protecting unpublished experimental data	Verify `is_admin` flag behavior in the `users` table; confirm that database-level role restrictions are correctly enforced; test that standard users cannot access administrative dashboard functions.
NFR6	Structured logging and error reporting for all ingestion events to support operational monitoring and auditability	Inspect `processed_files` table entries after ingestion runs; confirm that each event record captures file path, MD5 hash, data source, row count, file size, and timestamp; verify that skipped and rejected files are also logged with appropriate status.

It should be noted that two requirements, NFR3 and NFR4, are evaluated at the design level rather than through live testing. This is a deliberate and transparent methodological choice, consistent with DSR practice for quality attributes that are inherently difficult to measure quantitatively in early-stage research systems [31]. NFR3 (extensibility) is assessed through a structured analysis of the schema and ingestion architecture, while NFR4 (FAIR compliance) is addressed through a systematic mapping of system features to the FAIR principles, following the approach used in related RDM literature [10], [39].

6.1.3 Comparative Analysis

The second evaluation method situates the proposed DMS within the broader landscape of existing research data management solutions. The literature review presented in Chapter 3 identified a set of limitations and unaddressed gaps in established RDM systems, including OpenBIS and OMERO, particularly with respect to their applicability in heterogeneous, instrument-driven experimental research environments.

This comparative analysis examines the extent to which the proposed DMS addresses those specific gaps. Rather than performing a general feature comparison, the analysis is structured around the limitations identified in the literature: the absence of lightweight, domain-adaptable ingestion pipelines; tightly coupled architectures that hinder maintainability in small research groups; and insufficient support for heterogeneous, multi-instrument data integration without significant configuration overhead. For each identified gap, the corresponding design decision in the DMS is assessed for how effectively it resolves the limitation in the context of the Photosynthetic Microbes group, and more broadly for instrument-driven research groups.

The results of this comparative analysis are presented in Section 6.4.

6.2 System Evaluation

6.2.1 Functional Requirements Evaluation

FR1 — Heterogeneous data ingestion. The ingestion pipeline has been implemented and tested for all three instrument types across five data sources. Seven Micro-GC CSV files were processed end-to-end against the production database, yielding 6,090 rows in the `microgc_measurements` table across four gas variables (CO_2, O_2, N_2, C_2H_4). The PBR 1000L parser was validated against a real JSON file containing

1,431 records, producing 15,741 measurement rows in the `pbr_1000l_measurements` table. The PBR Meteorological and PBR Manual/Offline parsers were each validated against real instrument files through a dedicated test suite (`test_pbr_parsers.py`, all tests passed), confirming correct extraction of 75,592 rows from 19,080 tab-separated records and 508 rows from an 85-day, 7-sheet Excel workbook, respectively. A combined total of 20,816 measurement rows was confirmed across all populated tables. In each case, the MD5-based deduplication mechanism was validated by re-running the pipeline on an already-processed file and confirming that no duplicate records were inserted. FR1 is assessed as **Implemented**.

FR2 — Standardized naming conventions and contextual annotations. The controlled vocabulary mapping enforced at ingestion time was verified by querying the `variable_name` and `unit` columns of each measurement table across multiple ingestion runs from different instruments. All records conformed to the expected standardized labels defined in the mapping dictionaries, with no raw instrument-native labels present in the stored data. FR2 is assessed as **Implemented**.

FR3 — Centralized, cloud-accessible repository. The central PostgreSQL database instance is live and accessible via the connection credentials provisioned through CSC Pukki. The database was confirmed reachable from both the development environment and the deployed application hosted on CSC Pouta, using DBeaver as the database client for direct connection verification and schema inspection. The role-based access model was verified by confirming that the application role could retrieve stored data but was denied write access. In contrast, the admin role retained full privileges for ingestion and schema operations. FR3 is assessed as **Implemented**.

FR4 — Browsing and querying by experiment, instrument, date range, and researcher. The web application browsing and filtering functionality was verified in both the local development environment and the deployed instance on CSC Pouta, by executing filter queries against the `experiments` table across multiple

combinations of instrument type, organism, and date range, and confirming that results matched the expected records. The experiment detail view was verified by confirming that the contextual information stored in the `experiment_metadata` JSONB field was rendered correctly for selected experiments. FR4 is assessed as **Implemented**.

FR5 — Measurement visualization interface. The Dashboards view of the web interface was verified by confirming that charts were plotted correctly for all three instrument types. For the Micro-GC, measurement run charts displaying gas concentration values were verified against the corresponding records in the database. For the MIMS, continuous ion-current time-series charts were verified for accurate channel-to-variable mapping. For the PBR, multi-variable environmental parameter charts were verified against the ingested JSON snapshot data. In all cases, chart outputs matched the underlying database records for the selected experiment and variable. Researchers can select the experiment and variable of interest from controls within the interface, and the chart updates accordingly. FR5 is assessed as **Implemented**.

FR6 — Traceability of data origin, collection date, and applied transformations. The `processed_files` table was inspected after each ingestion run to confirm that the MD5 hash, source file path, ingestion timestamp, data source name, and inserted row count were recorded correctly for each processed file. The deduplication mechanism was verified by re-running the pipeline on a previously processed file and confirming that the pipeline logged a skip event and inserted no duplicate records. FR6 is assessed as **Implemented**.

6.2.2 Non-Functional Requirements Evaluation

NFR1 — Deployable within CSC infrastructure without external proprietary services. All system components operate entirely within CSC or open-source

infrastructure: the PostgreSQL database is hosted on CSC Pukki, the FastAPI backend and React frontend are deployed on CSC Pouta, and the ingestion pipeline runs on instrument-connected computers using open-source tooling. No external paid services are required for the core operation of the system. The open-source character of the toolchain (Python, PostgreSQL, FastAPI, React, Recharts) ensures that the system can be maintained and extended without licensing costs. NFR1 is assessed as **Implemented**.

NFR2 — Minimized manual effort through automated ingestion pipelines. The event-driven file system watcher has been configured for all ingestion scripts on the instrument-connected computers, triggering ingestion automatically when new files appear in monitored directories. The automated pipeline was verified by confirming that watcher-triggered invocations correctly detected new files, applied the deduplication check, and inserted records without manual intervention. A qualitative comparison with the previous workflow confirms a substantial reduction in manual steps: data that previously required manual file transfer, format conversion, and spreadsheet entry is now ingested and stored automatically. NFR2 is assessed as **Implemented**.

NFR3 — Extensible to support additional instrument types without schema changes. This requirement is evaluated at the design level through a structured analysis of the schema and ingestion architecture. The EAV measurement schema stores all instrument readings as typed rows of (`timestamp`, `variable_name`, `value`, `unit`), meaning that a new instrument type requires no alteration to existing measurement tables, only a new table inheriting from the shared SQLAlchemy base model. The `data_sources` table can register a new source by inserting a record without any schema modification. Adding ingestion support for a new instrument requires writing a new script following the established four-stage template, with no changes to the shared deduplication, validation, or insertion logic. This extensibility

was demonstrated in practice through the development of three structurally distinct ingestion scripts (Micro-GC, MIMS, PBR) without any schema modification between additions. NFR3 is assessed as **Design-level**.

NFR4 — Compliance with FAIR data principles within operational scope. The FAIR mapping established in Section 5.8 confirms that the system addresses Findability through structured, queryable experiment metadata; Accessibility through role-based web application access without requiring direct database credentials; Interoperability through standardized variable names, units, and open file formats; and Reusability through provenance records and JSONB preservation of raw instrument data [10]. As noted in Section 5.8, full FAIR compliance, including persistent identifier assignment and public catalogue registration, is beyond the current scope but is architecturally supported. NFR4 is assessed as **Design-level** [17].

NFR5 — Role-based access controls protecting unpublished data. The three-tier database role model (admin, application, researcher) was verified by confirming that the application role used by the web application could execute read queries but was denied write access. The `is_admin` flag behavior in the `users` table was confirmed by verifying that non-admin authenticated users could access experiment browsing and data export functions, but were denied access to administrative routes. NFR5 is assessed as **Implemented**.

NFR6 — Structured logging and error reporting for ingestion events. The `processed_files` audit table was inspected after a series of ingestion runs to confirm that every processed file generated a corresponding log record capturing the file path, file name, MD5 hash, data source name, file size in bytes, inserted row count, and ingestion timestamp. Skipped files, those whose MD5 hash matched an existing record, were confirmed to generate a log entry without inserting measurement rows. Files failing format validation were confirmed to generate an error log entry

without triggering a database transaction. This structured logging ensures that every ingestion event, whether successful, skipped, or failed, is traceable through the audit table. NFR6 is assessed as **Implemented**.

6.3 Ingestion Pipeline Performance and Error Handling

This section presents concrete evidence of system performance gathered during development and testing, including measurement row counts per instrument, dataset volumes, parser test outcomes, and edge cases encountered and resolved during implementation. This evidence supplements the requirements traceability assessment in Section 6.2 and directly supports the findings discussed in Section 6.5.

6.3.1 Measurement Row Counts and Dataset Sizes

End-to-end ingestion was validated across five data sources using real instrument files sourced from the group's UTU network drive. Table 6.3 summarizes the file counts, raw record counts, rows committed to or verified in each measurement table, and the variables captured per source.

Table 6.3: Measurement table row counts by instrument data source.

Instrument / Table	**Format**	**Files / Records**	**Rows**	**Variables**
Micro-GC	CSV†	7 files	6,090	CO_2, O_2, N_2, C_2H_4
PBR 1000L	JSON	1 file (1,431 records)	15,741	Dissolved O_2, temp., pH, CO_2 flow, LED intensity, harvest, density
PBR Meteorological	TXT	1 file (19,080 records)	75,592	Outdoor temp., indoor temp., radiation, daily radiation sum
PBR Manual/Offline	Excel	1 file (85 days, 7 sheets)	508	OD_{680}, OD_{750}, OD_{880}, dry wt, ash wt, N-NO_3, P-PO_4
Total			**20,816**	

† Not the proprietary format, instead, the summarized version of data extracted in CSV format from instrument software.

The ratio of raw records to extracted measurement rows reflects the wide-format reshape applied by each parser. The PBR 1000L parser reads 1,431 JSON records, each containing up to 11 simultaneous sensor readings, yielding 15,741 variable-level rows after normalization. The Meteorological parser reads 19,080 tab-separated rows, each with four sensor columns, producing 75,592 timestamped EAV rows after reshaping and timestamp correction. Raw source file sizes are modest: individual Micro-GC CSV files range from approximately 10 KB to 200 KB; the PBR 1000L JSON file is under 2 MB; the Meteorological TXT file covering 19,080 rows is approximately 1.5 MB; and the Manual/Offline Excel workbook spanning 85 experimental days across 7 sheets is under 500 KB. Aggregated across all tested files, the total raw

data volume is under 10 MB, yielding 20,816 database rows. As automated ingestion proceeds across all instruments over a multi-year operation period, table sizes are expected to grow substantially; range partitioning by month on the `timestamp` column is identified as a future schema optimization.

The `processed_files` registry recorded 7 entries at the close of the evaluation period, corresponding to the seven Micro-GC CSV files committed to the production database, each capturing the filename, MD5 content hash, data source identifier, row count, file size in bytes, and ingestion timestamp.

6.3.2 Parser Test Suite Results

The three PBR parsers were exercised against real instrument files through a dedicated test suite. Table 6.4 records the source file, raw record count, extracted measurement count, and date range confirmed per parser.

Table 6.4: PBR parser test suite results against real instrument files (all tests passed).

PBR Parser	**Source File**	**Raw Records**	**Rows Extracted**	**Date Range**
Online 1000L	Online PBR Data.json	1,431	15,741	2024-09-28 01:41–13:41
Meteorological Readings	Meteorological Data.txt	19,080	75,592	2025-04-08 to 2025-05-30
Offline Data	Offline Measurements.xlsx	85 days / 7 sheets	508	2025-04-08 to 2025-06-18

6.3.3 Edge Cases and Robustness Testing

A regression test suite was developed alongside the ETL pipeline to verify correct behavior under malformed, duplicate, and boundary-condition inputs. A total of 66 test assertions were executed across two test scripts covering parser correctness, upload deduplication, and file-detection logic; all passed. Table 6.5 summarizes the six principal categories of problematic input encountered during development, the data integrity risk each presented, and the outcome after correction.

Table 6.5: Principal input error categories encountered during development, associated data integrity risks, and outcomes after correction.

Input Category	**Data Integrity Risk**	**Outcome After Correction**	**Result**
Non-standard file format (Micro-GC CSV)	Instrument-specific delimiter pattern caused ingestion to fail silently rather than reject the file with a clear error	File correctly rejected at the validation stage with a descriptive log entry	PASS
Duplicate rows in instrument export	Instrument logging the same measurement twice caused values to be silently double-counted, producing scientifically incorrect stored results with no warning to the researcher	Duplicates detected and logged with affected rows identified; first occurrence retained	PASS

Continued on next page

Table 6.5 continued from previous page

Input Category	**Data Integrity Risk**	**Outcome After Correction**	**Result**
Re-upload of an already-ingested file	Uploading an identical file through the web interface under a different filename bypassed content-based deduplication and inserted duplicate measurement rows	Content hash used as the deduplication key regardless of filename or upload path	PASS
Unmapped records in manual data (PBR Excel)	Rows without a recognized date mapping silently received an incorrect fallback date, corrupting timestamps for affected measurements	Unmatched rows skipped with a logged warning; no fallback date assigned	PASS
Incorrect file detection pattern	A misconfigured file-matching pattern caused PBR JSON files to be silently ignored by the automated watcher, preventing ingestion without error	Pattern corrected; automated detection verified against representative files	PASS
Malformed timestamps in PBR data	Invalid timestamp strings were silently stored as null values in the database, making affected measurement rows unqueryable by time range	Invalid timestamps now raise a parsing error, preventing null storage and alerting the operator	PASS

The duplicate measurement case is worth particular attention from a data integrity

standpoint. Because the Micro-GC instrument occasionally logs the same injection twice in its raw export, the initial implementation silently produced incorrect stored values without any indication to the researcher. The correction ensures that whenever duplicates are present in a source file, a warning is logged that identifies the affected measurements, so the researcher is informed rather than receiving corrupted data silently. This illustrates a category of data quality risk that is unlikely to be detected consistently in manual spreadsheet workflows, and underlines the value of automated, validated ingestion over ad hoc file handling.

6.4 Comparative Discussion with Existing Solutions

The literature review in Chapter 3 identified a set of specific limitations in existing research data management systems, including OMERO, SEEK, and openBIS, which motivated the design of the proposed DMS. This section examines how the proposed system addresses each of those limitations directly.

6.4.1 Absence of Lightweight, Domain-Adaptable Ingestion Pipelines

A recurring limitation of existing systems identified in the literature review is that ingestion mechanisms are either tightly coupled to specific data formats or require significant configuration effort to adapt to new instrument types [20], [21]. OMERO is purpose-built for image data and provides no mechanism for ingesting tabular or JSON instrument streams without substantial custom development. SEEK and openBIS both support file attachment but do not provide structured ingestion pipelines that normalize and validate instrument data at the point of entry [25], [26].

The proposed DMS addresses this limitation through its modular, script-based ingestion architecture. Each ingestion script follows a common four-stage pipeline,

with instrument-specific parsing logic and a controlled vocabulary mapping encapsulated in a self-contained, independently testable unit. Adding support for a new instrument requires writing a new script following the established template, with no modification to the database schema or the shared pipeline infrastructure. This extensibility was demonstrated in practice through the development of three distinct ingestion scripts (Micro-GC, MIMS, PBR) that differ substantially in their source formats but share the same deduplication, validation, and insertion logic.

6.4.2 Tightly Coupled Architectures and Maintainability in Small Research Groups

Existing systems, such as openBIS and SEEK, are comprehensive platforms designed for large, multi-institutional deployments. Their breadth of functionality, while valuable in those contexts, introduces significant configuration and maintenance overhead for small research groups without dedicated IT support [25], [29]. Studies have noted that openBIS in particular requires substantial initial setup and ongoing administration, and that its steep learning curve limits adoption in groups where technical expertise is concentrated in one or two individuals [26].

The proposed DMS is designed with operational simplicity as a core principle. The use of a managed cloud database service eliminates server management responsibilities from the research group. The ingestion pipeline requires no framework installation beyond standard open-source libraries. The web application backend is organized into clearly separated route modules, and the frontend is a multi-page application with independent view components, making both layers straightforward to extend as requirements evolve. While the system was designed and evaluated in the context of the Photosynthetic Microbes group, this design philosophy, that a data management system must be operationally transparent to its users to achieve sustained adoption, applies equally to research groups of any size [7], [13].

6.4.3 Insufficient Support for Multi-Instrument Heterogeneous Data Integration

A third gap identified in the literature review concerns the difficulty of integrating data from multiple instruments with different output formats into a single coherent repository [18], [19]. Existing systems generally require either a uniform data format or format-specific plugins, both of which create friction when new instruments are added to a research workflow.

The EAV measurement schema of the proposed DMS resolves this by providing a format-agnostic storage layer: any instrument whose output can be reduced to timestamped variable-value pairs can be accommodated without schema changes. The `data_sources` table decouples the logical description of a data stream from its physical ingestion format, and the `experiment_metadata` JSONB column provides extensible storage for instrument- and experiment-specific context that does not fit the relational schema [3]. This architecture enabled the integration of three structurally distinct instrument types, CSV with fixed columns (Micro-GC), CSV with variable channel labels (MIMS), and JSON snapshots (PBR), within a single unified schema, without any schema modification between instrument additions.

6.4.4 Summary

Table 6.6 summarizes the comparative assessment of the proposed DMS against the three existing systems evaluated in the literature review, with respect to the key gaps identified above.

Table 6.6: Comparative assessment of the proposed DMS against selected existing RDM systems with respect to the gaps identified in the literature review.

Criterion	**OMERO**	**SEEK**	**openBIS**	**Proposed DMS**
Lightweight ingestion pipeline for heterogeneous instrument data	No	Partial	Partial	Yes
Adaptable to new instrument types without schema changes	No	No	No	Yes
Low configuration and maintenance overhead for research groups	No	Partial	No	Yes
Multi-format data integration in a unified schema	No	Partial	Partial	Yes
Deployable within institutional cloud infrastructure	Partial	Partial	Partial	Yes
Open-source, no licensing costs	Yes	Yes	Yes	Yes

6.5 Observations and Findings

The evaluation presented in this chapter supports the following principle observations.

The unified measurement EAV schema is effective for heterogeneous multi-instrument data integration. The evaluation confirms that a schema built around a shared structure of timestamped variable-value pairs with a shared structure across instrument types can accommodate heterogeneous data sources without requiring structural modifications as new sources are added. This finding directly addresses one of the persistent gaps identified in the literature, the absence of flexible, instrument-independent storage designs in existing RDM systems [36].

Automating the data collection process meaningfully reduces research overhead and data quality risk. The evaluation demonstrates that replacing manual file transfer with an event-driven ingestion mechanism eliminates the steps previously required to transfer, format, and record instrument output, which the literature identifies as primary sources of transcription error and provenance loss in experimental research settings [2], [18]. The built-in deduplication mechanism further strengthens data integrity without placing additional demands on the users.

The devised solution is operationally simpler than existing solutions for the target context. The comparative analysis confirms that the proposed DMS addresses the three principal gaps identified in the literature review, ingestion adaptability, operational simplicity, and multi-format integration, more directly than OMERO, SEEK, or openBIS in the context of an instrument-driven photosynthesis research. This comes with the acknowledged trade-off that the proposed system offers a narrower feature set; it does not provide the collaborative annotation tools of SEEK, the image analysis integration of OMERO, or the laboratory information management features of openBIS, and reflects the broader finding that fitness for purpose is a more consequential selection criterion than feature breadth in research data management [7].

FAIR compliance is partial and architecturally extensible. The system satisfies the Findable, Accessible, and Reusable dimensions of FAIR within its operational scope, and partially satisfies Interoperability through standardized naming and open formats. Full FAIR compliance, particularly persistent identifier assignment and public metadata publication, is not yet achieved, but the system's architecture has been deliberately designed to support this extension without requiring structural changes [10], [17].

7 Discussion

7.1 Interpretation of Results

The evaluation results presented in Chapter 6 indicate that the proposed DMS fulfills its stated functional and non-functional requirements within the operational scope defined in Chapter 4. All six functional requirements were verified against real experimental data, confirming that the system can collect, normalize, store, query, visualize, and trace heterogeneous instrument-generated data within a unified relational database. The automated ingestion pipeline, driven by an event-based file system watcher, reduces the fragmented manual workflows that previously characterized data handling in the group, and the built-in deduplication mechanism supports data integrity without requiring user intervention.

The results also indicate that the design choices made in Chapter 5 translated correctly into implemented behavior. The format-agnostic measurement schema, combined with a controlled vocabulary layer enforced at the point of ingestion, accommodated heterogeneous, structurally distinct data sources across all three instrument types without requiring schema changes between instrument additions, demonstrating the practical viability of this design approach for instrument-driven research environments [3]. The decision to preserve original instrument records alongside normalized values ensured that the normalization process remains auditable, a property that the literature identifies as important for long-term reproducibility in

experimental research [40]. The layered access model, with distinct database roles for the ingestion pipeline and the application tier, functioned as intended during testing.

The proposed system also met the data access and visualization requirements, confirming that researchers can access and explore their experimental data through the web interface without database interaction. The ability to select experiments and variables from controls within the same interface used for data browsing is consistent with findings in the RDM literature that usability and low interaction cost are important determinants of sustained system adoption in research settings [25], [29].

The two quality requirements assessed at the design level, extensibility and FAIR alignment, were evaluated through structured analysis rather than live testing. This is a deliberate methodological choice consistent with DSR practice for quality attributes that are inherently difficult to measure quantitatively in early-stage systems [31]. The structured analysis conducted in Chapter 6 supports the conclusion that the system's schema and modular ingestion architecture are well-positioned to accommodate new instrument types without changes to the core database schema, though it is acknowledged that new instruments may require updates to ingestion scripts, metadata fields, validation rules, or visualization configuration depending on the characteristics of the new data source. The system's data architecture also provides a foundation for future FAIR compliance work without requiring fundamental redesign.

7.2 Answers to Research Questions

RQ-1: How can heterogeneous, instrument-generated experimental data streams be structured, cleaned, and standardized to ensure usability, reproducibility, and long-term accessibility?

The findings suggest that a combination of a unified, instrument-independent measurement schema, a controlled vocabulary for variable names and units enforced

at ingestion, and a hash-based deduplication mechanism constitutes an effective approach to the structuring and standardization challenge in the context of this study. The schema design decouples the logical representation of measurements from the physical format of instrument outputs, allowing the same structure to accommodate heterogeneous sources including CSV tabular data (Micro-GC, MIMS), JSON snapshot data (PBR PhycoFlow), tab-separated TXT (Meteorological), and Excel-based offline records, without structural modification as new sources are added [36]. The controlled vocabulary mapping at ingestion enforces terminological consistency, ensuring that stored data remains interpretable regardless of which instrument produced it, and directly addressing the terminological fragmentation that the literature identifies as a recurring obstacle to multi-instrument data reuse [18]. The preservation of original unparsed records alongside normalized values ensures that processing decisions are auditable, supporting the reproducibility requirements of experimental research [40].

Accessibility is provided through a web interface that allows researchers to browse, visualize, and retrieve their data without database interaction, complemented by a REST API for integration with external analysis tools. Together, these address the accessibility dimension of FAIR within the system's operational scope [10].

RQ-2: Which data management architectures and design principles best support the storage, integration, and retrieval of multi-format experimental data in a collaborative research environment?

The findings suggest that a layered architecture combining a relational storage backend with a modular, event-driven ingestion layer and a web-based access interface is well-suited to the requirements of an instrument-driven experimental research group operating within an institutional cloud infrastructure. The hybrid schema strategy, typed relational columns for structured queryable attributes alongside JSONB columns for contextual metadata and provenance, proved capable of accommodating

the varied and evolving data structures of an active research group without requiring structural revision as the system developed [3]. The hierarchical organization of the data model supports natural query scoping at any level of granularity while maintaining referential integrity.

The modular ingestion architecture supports extensibility: new instruments can be integrated by adding new ingestion scripts without modifying the shared pipeline infrastructure or the database schema. It should be noted, however, that this extensibility applies specifically to the core schema and shared pipeline logic; individual instruments may still require tailored parsing, metadata configuration, and visualization handling. This partially addresses the extensibility limitations documented for existing RDM systems such as OMERO and OpenBIS [21], [26].

The deployment of the system across institutional cloud infrastructure, the database on CSC Pukki and the application on CSC Pouta, confirms that institutional cloud services can serve as a viable, low-overhead hosting platform for research data systems, eliminating the need for dedicated server administration by the research group [37].

More broadly, the findings suggest that architectural simplicity and operational transparency are important selection criteria when deploying a data management system in a research environment, particularly for small groups without dedicated IT support. A system that researchers can use effectively without needing to understand its underlying technical implementation is more likely to achieve sustained adoption than a feature-rich system that requires specialist administration [7], [13].

7.3 Generalization to Other Research Domains

Although the proposed DMS was designed and evaluated exclusively in the context of the Photosynthetic Microbes research group at the University of Turku, the architectural principles that underpin the system are not domain-specific and

may be applicable to other instrument-driven experimental research settings. This transferability is argued at the level of design principles rather than verified through external testing, and should be understood as a hypothesis for future evaluation rather than a demonstrated outcome.

The core requirement the system was designed to meet, integrating heterogeneous instrument outputs into a unified queryable repository with minimal manual handling, is common across many experimental disciplines, including environmental monitoring, analytical chemistry, bioreactor-based bioprocess engineering, and materials characterization. In each of these fields, research groups routinely operate multiple instruments with distinct output formats and rely on manual procedures for data transfer that introduce quality risks [2], [18].

The database schema, controlled vocabulary mapping approach, and modular ingestion architecture are instrument-agnostic by design. Any data stream that can be reduced to timestamped variable-value pairs, a description that encompasses a broad range of continuous measurement instruments, can in principle be integrated into the system by developing an appropriate ingestion script. The experiment-level data model, combining structured relational attributes with flexible JSONB fields for contextual annotation, is similarly adaptable across different experimental paradigms [12].

The deployment model, using CSC's managed PostgreSQL service and open-source frameworks, is replicable at any Finnish research institution with access to CSC services, and the general approach is transferable to analogous institutional cloud platforms elsewhere. The reliance exclusively on open-source tooling ensures that no licensing barriers restrict adoption [8].

It should be noted that the system has been validated only within one research group and one experimental domain. Whether the architecture performs as expected, scales appropriately, and meets usability expectations in other contexts has not been

tested and remains an open question. Research domains with substantially different data characteristics, such as genomics, medical imaging, or social science survey data, would require significant adaptation and may be better served by domain-specific platforms [15].

7.4 Challenges and Limitations of the Proposed System

Several challenges and limitations of the proposed system should be acknowledged.

FAIR compliance is partial. While the system addresses the Findable, Accessible, and Reusable dimensions of FAIR within its operational scope, full FAIR compliance, including the assignment of globally persistent identifiers to datasets and the publication of machine-readable metadata in a public catalogue, is not yet achieved. The addition of DOI assignment through a service such as DataCite and registration with an institutional data catalogue would be required to close this gap [10], [17].

No semantic interoperability. The controlled vocabulary used for variable names and units is an internal, informally defined standard, not formally aligned with established scientific ontologies such as ChEBI or UO. This limits cross-system data integration and would need to be addressed before the system could contribute data to federated research data platforms [16].

Single-group scope. The system has been designed and evaluated for the Photosynthetic Microbes group specifically. Multi-tenancy is not currently supported. The `groups` table provides a foundation for this extension, but access control, data isolation, and schema management for multi-group operation would require additional development.

Experiment entity limited to single instrument per record. As noted

in Section 5.3, each experiment record in the current data model links to a single instrument and data source. In biological experiments where multiple instruments are used simultaneously, the resulting datasets are stored as separate experiment records. Handling the association between co-occurring instrument datasets from a single biological experiment at the query level is identified as a direction for future development.

Ingestion constraints and institutional boundaries. Automated ingestion depends on the file system watcher running on instrument-connected computers with stable network access to the database. Interruptions in network connectivity or instrument computer availability will cause missed ingestion events. A more robust architecture would incorporate a queuing mechanism or a persistent ingestion agent. Additionally, the system was developed within the constraints of institutionally approved, open-source, and free-to-use services provided by CSC, which, while appropriate for an academic deployment, limits infrastructure choices and may require adaptation if deployed outside the Finnish academic context.

Evaluation scope. The evaluation conducted in this thesis is limited to implementation-level verification against real experimental data. Long-term performance, robustness under continuous use, scalability under larger data volumes, and practical usability from the perspective of the research group members have not been formally assessed and constitute important directions for future evaluation.

7.5 Future Work

Several directions for future development of the DMS are identified on the basis of the evaluation findings and the limitations noted above.

Persistent identifier integration and public metadata publication. The most significant step toward full FAIR compliance would be the assignment of DOIs to individual datasets via a service such as DataCite and the registration of dataset

metadata in the University of Turku's institutional data catalogue or a domain-specific repository such as Zenodo. The structured `experiments` table provides a natural source for the metadata record required for DOI registration [9].

Ontology alignment for the controlled vocabulary. Aligning the controlled vocabulary with established scientific ontologies, particularly ChEBI for chemical species and UO for measurement units, would improve semantic interoperability and enable cross-system data integration. This would involve extending the mapping dictionaries to include ontology term URIs alongside the standardized labels [16].

Multi-instrument experiment association. Future development should address the current limitation that each experiment record links to a single instrument. A many-to-many relationship between biological experiments and instrument datasets would allow co-occurring measurements from multiple instruments to be explicitly associated at the data model level, improving downstream querying and analysis for complex multi-instrument experiments.

Extension of the REST API and programmatic data access. The existing FastAPI REST API could be extended with more granular query parameters, paginated bulk export, and integration with notebook-based analysis environments such as Jupyter, further reducing friction for researchers who prefer programmatic data access [3].

Raw file archival to CSC Allas. The planned integration pathway involving periodic upload of raw instrument files to CSC Allas object storage [44] would extend the provenance chain beyond the database's JSONB fields to the original binary source files, providing geographically redundant institutional archival of primary data.

Extension to additional instruments and multi-group deployment. As the Photosynthetic Microbes group acquires new instruments or as the DMS is considered for adoption by additional research groups, the ingestion architecture can

be extended by adding new scripts and data source records. Multi-group deployment would require implementation of group-level data isolation in the access control layer and a shared administration interface for group account management.

Structured usability evaluation. A structured evaluation of the system's usability from the perspective of group researchers, including task-based testing and feedback collection, would provide empirical evidence of the system's practical effectiveness in routine research workflows and would inform future interface and workflow improvements.

8 Conclusion

This thesis designed, implemented, and evaluated a research data management system that addresses the fragmented, manual data handling workflows characterizing instrument-driven experimental research. Motivated by the specific challenges of the Photosynthetic Microbes research group at the University of Turku, where heterogeneous outputs from the Micro-GC, MIMS, and PBR instruments were managed through local file storage, spreadsheets, and informal conventions with no unified mechanism for structured storage, automated ingestion, or systematic contextual annotation, the work produced a functioning system with automated ingestion, centralized cloud storage, a structured data model linking experimental context to individual measurements, and a web-based interface for data browsing and visualization.

Research Question 1 asked how heterogeneous, instrument-generated experimental data streams can be structured, cleaned, and standardized to ensure usability, reproducibility, and long-term accessibility. The findings indicate that a unified, instrument-independent Entity-Attribute-Value measurement schema, combined with a controlled vocabulary for variable names and units enforced at the point of ingestion, provides a workable approach to this challenge within the scope of the study. This approach accommodated structurally distinct instrument outputs within a single schema without requiring schema changes as new data sources were added, and supports the long-term reproducibility of experimental findings through provenance

preservation. Accessibility is provided through a web interface that researchers can use without database interaction, supported by a REST API for integration with external tools. It is acknowledged that the scalability and generalizability of this approach beyond the current context have not been formally tested.

Research Question 2 asked which data management architectures and design principles best support the storage, integration, and retrieval of multi-format experimental data in a collaborative research environment. The evaluation suggests that a layered architecture combining a structured relational storage backend, a modular event-driven ingestion layer, and a unified web-based access interface, deployed on institutional cloud infrastructure, is well-suited to the needs of an instrument-driven research group of this type. The comparative analysis indicates that the system addresses the three principal gaps identified in the literature review, ingestion adaptability, operational simplicity, and multi-format integration, more directly than established general-purpose platforms in the context of this group. The findings reinforce the broader observation that fitness for purpose and operational transparency are important design criteria in research data management contexts, though this conclusion is drawn from a single-group evaluation and would benefit from validation in broader settings.

The system was evaluated against six functional and six non-functional requirements derived from the design science research framework. All six functional requirements were verified against real experimental data. Four of the six non-functional requirements were confirmed through live testing, and the remaining two, extensibility and FAIR compliance, were assessed through structured design-level analysis, consistent with DSR evaluation practice for quality attributes that are difficult to measure quantitatively in early-stage systems. The evaluation is acknowledged to be at the implementation level; long-term performance, robustness under continuous use, and practical usability from the perspective of group researchers remain to be

assessed.

The principal contributions of this thesis are: a validated design for a lightweight, domain-appropriate RDM system suited to instrument-driven experimental research; a modular ingestion architecture that accommodates new instrument types without changes to the core database schema, though new instruments may require tailored parsing, metadata, and visualization handling; a FastAPI REST API enabling programmatic data access; and a demonstration that managed institutional cloud infrastructure can serve as a viable, low-overhead hosting platform for research database systems in an academic context. Identified directions for future work include persistent identifier integration for full FAIR compliance, ontology alignment for the controlled vocabulary, multi-instrument experiment association at the data model level, structured usability evaluation with group researchers, extension of the REST API with additional query capabilities, and multi-group deployment support.

Research data management is increasingly recognized as a structural challenge for the scientific community, not merely a technical one [7]. The system presented in this thesis does not resolve that challenge at the institutional level, but it demonstrates that a thoughtfully designed, domain-appropriate DMS, built from open-source components and deployed on existing institutional infrastructure, can meaningfully improve the data practices of a research group within the constraints of an academic project. The design principles documented here may be transferable to other instrument-driven research settings facing similar challenges, and provide a concrete foundation for future extension toward full FAIR compliance and open science practice [10].

References

[1] T. Cunha-Oliveira, J. P. A. Ioannidis, and P. J. Oliveira, "Best practices for data management and sharing in experimental biomedical research", *Physiological Reviews*, vol. 104, no. 3, pp. 1387–1408, 2024. DOI: 10.1152/physrev.00043.2023.

[2] C. Tenopir et al., "Data sharing by scientists: Practices and perceptions", *PLOS ONE*, vol. 6, no. 6, e21101, 2011. DOI: 10.1371/journal.pone.0021101.

[3] M. Kleppmann, *Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems*. Sebastopol, CA, USA: O'Reilly Media, 2017.

[4] T. Hey, S. Tansley, and K. Tolle, Eds., *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Redmond, WA, USA: Microsoft Research, 2009, ISBN: 978-0-9825442-0-4.

[5] M. Armbrust, A. Ghodsi, R. Xin, and M. Zaharia, "Lakehouse: A new generation of open platforms that unify data warehousing and advanced analytics", in *Proceedings of the 11th Annual Conference on Innovative Data Systems Research (CIDR '21)*, Online, Jan. 2021. [Online]. Available: https://www.cidrdb.org/cidr2021/papers/cidr2021_paper17.pdf.

[6] L. Hsu, R. L. Martin, B. McElroy, K. Litwin-Miller, and W. Kim, "Data management, sharing, and reuse in experimental geomorphology: Challenges,

strategies, and scientific opportunities", *Geomorphology*, vol. 244, pp. 180–189, 2015. DOI: 10.1016/j.geomorph.2015.03.039.

[7] C. L. Borgman, "The conundrum of sharing research data", *Journal of the American Society for Information Science and Technology*, vol. 63, no. 6, pp. 1059–1078, 2012. DOI: 10.1002/asi.22634.

[8] S. Higgins, "The DCC curation lifecycle model", *International Journal of Digital Curation*, vol. 3, no. 1, pp. 134–140, 2008. DOI: 10.2218/ijdc.v3i1.48.

[9] J. Lehmann, S. Schorz, A. Rache, T. Häußermann, M. Rädle, and J. Reichwald, "Establishing reliable research data management by integrating measurement devices utilizing intelligent digital twins", *Sensors*, vol. 23, no. 1, p. 468, 2023. DOI: 10.3390/s23010468.

[10] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, et al., "The FAIR guiding principles for scientific data management and stewardship", *Scientific Data*, vol. 3, p. 160 018, 2016. DOI: 10.1038/sdata.2016.18.

[11] D. R. Donaldson and J. W. Koepke, "A focus groups study on data sharing and research data management", *Scientific Data*, vol. 9, no. 1, p. 345, 2022. DOI: 10.1038/s41597-022-01428-w.

[12] N. Preuss, G. Staudter, M. Weber, R. Anderl, and P. F. Pelz, "Methods and technologies for research and metadata management in collaborative experimental research", *Applied Mechanics and Materials*, vol. 885, pp. 170–183, 2018. DOI: 10.4028/www.scientific.net/amm.885.170.

[13] L. Perrier, E. Blondal, A. P. Ayala, D. Dearborn, T. Kenny, D. Lightfoot, R. Reka, M. Thuna, L. Trimble, and H. MacDonald, "Research data management in academic institutions: A scoping review", *PLOS ONE*, vol. 12, no. 5, e0178261, 2017. DOI: 10.1371/journal.pone.0178261.

[14] V. Navale and M. McAuliffe, "Long-term preservation of biomedical research data", *F1000Research*, vol. 7, no. 1, p. 1353, 2018. DOI: 10.12688/f1000research.16015.1.

[15] A. Lefebvre and M. Spruit, "Laboratory forensics for open science readiness: An investigative approach to research data management", *Information Systems Frontiers*, vol. 25, no. 1, pp. 381–399, 2021. DOI: 10.1007/s10796-021-10117-5.

[16] E. Alharbi, R. Skeva, N. Juty, C. Jay, and C. Goble, "Exploring the current practices, costs and benefits of FAIR implementation in pharmaceutical research and development: A qualitative interview study", *Data Intelligence*, vol. 3, no. 4, pp. 507–527, 2021. DOI: 10.1162/dint_a_00109.

[17] A. Jacobsen et al., "FAIR principles: Interpretations and implementation considerations", *Data Intelligence*, vol. 2, no. 1–2, pp. 10–29, 2020. DOI: 10.1162/dint_r_00024.

[18] D. Mittal, R. A. Mease, T. Kuner, H. Flor, R. Kuner, and J. Andoh, "Data management strategy for a collaborative research center", *GigaScience*, vol. 12, giad049, 2022. DOI: 10.1093/gigascience/giad049.

[19] A. Stoewer et al., "Web data storage for management and sharing of neuroscience data", *Frontiers in Neuroinformatics*, vol. 7, p. 17, 2013. DOI: 10.3389/fninf.2013.00017.

[20] M. Cuellar-Friedrich et al., "A data management infrastructure for the integration of imaging and omics data in life sciences", *BMC Bioinformatics*, vol. 23, no. 1, p. 1, 2022. DOI: 10.1186/s12859-021-04556-z.

[21] A.-S. Jannasch et al., *Setting up an institutional OMERO environment for bioimage data: Perspectives from both facility staff and users*, Preprint, bioRxiv, Accessed: Mar. 2026, 2024. DOI: 10.1101/2024.01.18.576169.

[22] C. Allan et al., "OMERO: Flexible, model-driven data management for experimental biology", *Nature Methods*, vol. 9, no. 3, pp. 245–253, 2012. DOI: 10.1038/nmeth.1896.

[23] K. Wolstencroft et al., "SEEK: A systems biology data and model management platform", *BMC Systems Biology*, vol. 9, p. 33, 2015. DOI: 10.1186/s12918-015-0174-y.

[24] D. Pradhan, H. Ding, J. Zhu, B. P. Engelward, and S. S. Levine, "NExtSEEK: Extending SEEK for active management of interoperable metadata", *Journal of Biomolecular Techniques*, vol. 33, no. 1, 3fc1f5fe.db404124, 2022. DOI: 10.7171/3fc1f5fe.db404124.

[25] Y. Hemani, K. Koch, O. Mendo-Diaz, S. Bachhofner, S. Baffelli, and D. Bleiner, "The openBIS digital platform for instrumentation and data workflow in the analytical laboratory", *CHIMIA*, vol. 79, no. 1–2, pp. 36–45, 2025. DOI: 10.2533/chimia.2025.36.

[26] F. Plass, S. Englisch, B. Apeleo Zubiri, L. Pflug, E. Spiecker, and M. Stingl, "Using OpenBIS as virtual research environment: An ELN-LIMS open-source database tool as a framework within the CRC 1411 design of particulate products", *Data Science Journal*, vol. 22, no. 1, p. 3, 2023. DOI: 10.5334/dsj-2023-003.

[27] A. Bauch et al., "openBIS: A flexible framework for managing and analyzing complex data in biology research", *BMC Bioinformatics*, vol. 12, no. 1, p. 468, 2011. DOI: 10.1186/1471-2105-12-468.

[28] K. Schweinar, F. Wagner, C. M. Klingner, S. Festag, C. Spreckelsen, and S. Brodoehl, "Simplifying multimodal clinical research data management: Introducing an integrated and user-friendly database concept", *Applied Clinical Informatics*, vol. 15, no. 2, pp. 234–249, 2024. DOI: 10.1055/a-2259-0008.

[29] A. M. Mukhin, F. V. Kazantsev, and S. A. Lashin, "Laboratory information systems for research management in biology", *Vavilov Journal of Genetics and Breeding*, vol. 27, no. 7, pp. 898–905, 2023. DOI: 10.18699/VJGB-23-25.

[30] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design science in information systems research", *MIS Quarterly*, vol. 28, no. 1, pp. 75–105, 2004. DOI: 10.2307/25148625.

[31] A. Hevner and S. Chatterjee, *Design Research in Information Systems: Theory and Practice* (Integrated Series in Information Systems). New York, NY, USA: Springer, 2010, vol. 22. DOI: 10.1007/978-1-4419-5653-8.

[32] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research", *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45–77, 2007. DOI: 10.2753/MIS0742-1222240302.

[33] B. P. Regmi and M. Agah, "Micro gas chromatography: An overview of critical components and their integration", *Analytical Chemistry*, vol. 90, no. 22, pp. 13 133–13 150, 2018. DOI: 10.1021/acs.analchem.8b01461.

[34] A. Burlacot, F. Burlacot, Y. Li-Beisson, and G. Peltier, "Membrane inlet mass spectrometry: A powerful tool for algal research", *Frontiers in Plant Science*, vol. 11, p. 1302, 2020. DOI: 10.3389/fpls.2020.01302.

[35] S. N. Chanquia, G. Vernet, and S. Kara, "Photobioreactors for cultivation and synthesis: Specifications, challenges, and perspectives", *Engineering in Life Sciences*, vol. 22, no. 12, pp. 712–724, 2022. DOI: 10.1002/elsc.202100070.

[36] M. Fowler, *Patterns of Enterprise Application Architecture*. Boston, MA, USA: Addison-Wesley, 2002.

[37] CSC — IT Center for Science, *Pukki: Database-as-a-service (DBaaS)*, https://research.csc.fi/service/pukki-database-as-a-service-dbaas/, Accessed: Mar. 2026, 2024.

[38] W. McKinney, *Python for Data Analysis: Data Wrangling with pandas, NumPy, and Jupyter*, 3rd ed. Sebastopol, CA, USA: O'Reilly Media, 2022.

[39] H. S. Rzepa and A. P. Tonge, "A metadata-driven approach to data repository design", *Journal of Cheminformatics*, vol. 9, no. 1, p. 6, 2017. DOI: 10.1186/s13321-017-0190-6.

[40] F. Linnebank et al., "From bench to brain: A metadata-driven approach to research data management in a collaborative neuroscientific research center", *Data Science Journal*, vol. 24, no. 1, p. 2, 2025. DOI: 10.5334/dsj-2025-002.

[41] J. Densmore, *Data Pipelines Pocket Reference: Moving and Processing Data for Analytics*. Sebastopol, CA, USA: O'Reilly Media, 2021.

[42] T. Bray, "The JavaScript object notation (JSON) data interchange format", Internet Engineering Task Force, RFC 8259, 2017. DOI: 10.17487/RFC8259.

[43] Recharts Team, *Recharts: Redefined chart library built with React and D3*, Accessed: 2024, 2024. [Online]. Available: https://recharts.org.

[44] CSC — IT Center for Science, *Allas object storage service description*, https://research.csc.fi/allas-service-description/, Accessed: Mar. 2026, 2024.