ODELIA

Open Consortium for Decentralized Medical Artificial Intelligence

HORIZON-HLTH-2021-CARE-05-02

Deliverable D4.1

A web front-end with basic functionalities

Lead beneficiary	SAI
Author(s)	David Friedrich – Product Management Bulat Maksudov – Architecture, development, testing, user training
Reviewer(s)	Alice Dudle (UZH)
Dissemination level	PU
Type	DEC
Delivery date	June 30, 2025

ODELIA is funded by the European Union's Horizon Europe Framework under Grant Agreement 101057091

Funded by the European Union

TABLE OF CONTENTS

ODELIA

DISCLAIMER

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

EXECUTIVE SUMMARY

We report the successful completion of Deliverable 4.1 (Web-front end with basic functionalities). This milestone provides ODELIA consortium members with a user-friendly web-based MRI viewer designed to

- Collect manual annotations to support the development of new AI models.
- Submit MRI studies for AI-based automated breast cancer diagnosis
- Display AI-generated diagnostic results along with the underlying reasoning

Our viewer enables the convenient classification of breast cancer lesions using AI models developed by the ODELIA consortium. It also facilitates easy review of the results to further improve model performance.

INTRODUCTION

The objective of WP4 is to develop a user-friendly front-end interface that facilitates both the efficient collection of data for new AI model development and the testing of these models in clinical workflows. As a first step, the completion of Deliverable D4.1 provides the consortium members with a web front-end with basic functionalities designed to

- Collect manual annotations to support the development of new AI models.
- Submit MRI studies for AI-based automated breast cancer diagnosis
- Display AI-generated diagnostic results along with the underlying reasoning

Our work builds on the preliminary efforts carried out by OSIMIS, a former consortium partner that was replaced by StratifAI in early 2025 following OSIMIS's bankruptcy. StratifAI's deliverable already includes a prototype back-end AI processing pipeline capable of running and displaying results from an AI model trained at UKA. Consequently, the viewer functionality exceeds the requirements of Deliverable 4.1, which only mandated the display of precomputed model results on a fixed dataset.

This report is structured as follows: Section 1 and 2 outline the general requirements, design choices and frameworks we use. Section 3 details the workflow of the radiologist interacting with our system. Sections 4-7 describes the core functionalities implemented for this deliverable, namely Launching the viewer, the labeling functionality, the Sent to AI functionality and the display of AI-generated results within the viewer.

The source code of the viewer (Deliverable D4.2) is published alongside this report in the Achievements section of the ODELIA consortium website.

SECTION 1: VIEWER REQUIREMENTS

As a first step, we conducted a rapid yet comprehensive review of the requirements and priorities to ensure alignment with project goals and timelines, taking into account the time lost due to the changes in the lead partner of WP 4. To this end, we worked closely with the ODELIA Partners from UKA (Univ.-Prof. Dr. med. Dipl.-Phys Daniel Thrun, Dr. Gustav Müller-Franzes) to validate use case priorities and ensure that our efforts are focused on delivering the highest value in the shortest possible time for the final users of the Viewer. To solidify this direction and ensure consortium-wide alignment, we presented our proposed approach and received confirmation during the April monthly call with the ODELIA consortium to establish a shared understanding of priorities and timelines.

The final MRI Viewer will provide the following features and functionality:

ID	Feature	Functionality
ODV-001	Launch ODELIA web viewer	launch a web viewer, select particular MRI examination from a list and of studies, review it in main view, do basic manipulations with the images.
ODV-002	Provide manual annotation tools	Tools for annotating lesions within the MRI for AI model training.
ODV-003	Send to AI	Select AI model from list of available AI models, add new models, send an MRI examination for processing by AI, enable user to select the correct sequences for AI inference.
ODV-004	View AI results	Select AI results produced from different algorithms, display classification results and confidence, and explainability maps (if provided by the model).
ODV-005	Provide Feedback on AI results	Feedback interface for the standardized collection of feedback of the AI model performance.
ODV-006	Show Feedback results	Customizable dashboard to visualize feedback results.
ODV-007	Enable User authentication	Secure user authentication, e.g. to link reviewer name to the feedback provided.

For deliverable D4.1, we have addressed first implementations of features ODV-001 till ODV-004.

SECTION 2: DESIGN CHOICES AND DEVELOPMENT FRAMEWORK

Based on the agreed upon requirements for the ODELIA Viewer, we re-assessed the design choices, architectural overview and established a fit-for-purpose development methodology.

Design Choices

In general, we build on and confirmed the Design choices taken by OSIMIS. In summary, these are:

- DICOM interoperability. Taking into account the recent surge in Medical AI advances and the formation of DICOM WG-23 (https://www.dicomstandard.org/activity/wgs/wg-23), which focuses on interaction between Radiological systems and AI, we opted for DICOM interoperability as our guiding principle during the development process.
- Following the spirit of open-source development, we selected a list of FOSS Radiological viewers, which we considered as the base for ODELIA Viewer development. Among those options we decided to adopt OHIF as our base with our considerations being as follows:
    1. Permissive license, which allows us to modify and redistribute the source code to comply with the funding requirements.

2. Large community for us to rely on for guidance and support, while simultaneously improving research dissemination and wider adoption.
3. Emergent interest of AI researchers in using OHIF as basis for labeling and AI integration, such as IDC (https://datacommons.cancer.gov/repository/imaging-data-commons) and MONAILabel (https://github.com/Project-MONAI/MONAILabel).

Architecture Overview

This section provides a high-level description of the ODELIA Viewer architecture, supported by Container and Component diagrams. These diagrams illustrate the system's structure, interactions, and internal organization.

The Container diagram (Figure 1) presents the key systems and their interactions: The radiologist is the primary user interacting with the ODELIA Viewer web application. The viewer handles image display and user interface logic. The viewer communicates with the Orthanc Viewer, which functions as the DICOM server managing image storage and retrieval. The viewer queries image data and submits studies for AI processing using DICOMweb/REST. The Orthanc Viewer routes studies to Orthanc AI, a DICOM processing service that interacts with the AI Backend for inference requests via REST/HTTP. The AI Backend returns analysis results to Orthanc AI, which wraps them into DICOM format and sends them back to Orthanc Viewer. Finally, Orthanc Viewer delivers the AI results to the ODELIA Viewer, which displays them to the radiologist. Communication between components strictly uses standard protocols (DICOMweb/REST, REST/HTTP), ensuring interoperability and modularity.

The Component diagram (Figure 2) breaks down the ODELIA Viewer container into internal modules and extensions. The core OHIF Viewer UI handles image rendering and user interactions. The DICOMweb Client manages communication with Orthanc Viewer via DICOMweb and REST.

The Extension Manager oversees the lifecycle and integration of three main OHIF extensions:

- **Send to AI extension**: manages study submission workflows to Orthanc Viewer for AI processing.
- **View AI Results extension**: retrieves AI annotations and displays them within the viewer.
- **Labeling extension**: provides manual annotation tools to support data collection.

This breakdown clarifies data flow and responsibilities inside the viewer and shows how modular extensions enhance the core functionality. The design ensures separation of concerns and maintainability, facilitating integration of additional AI-related features in the future.

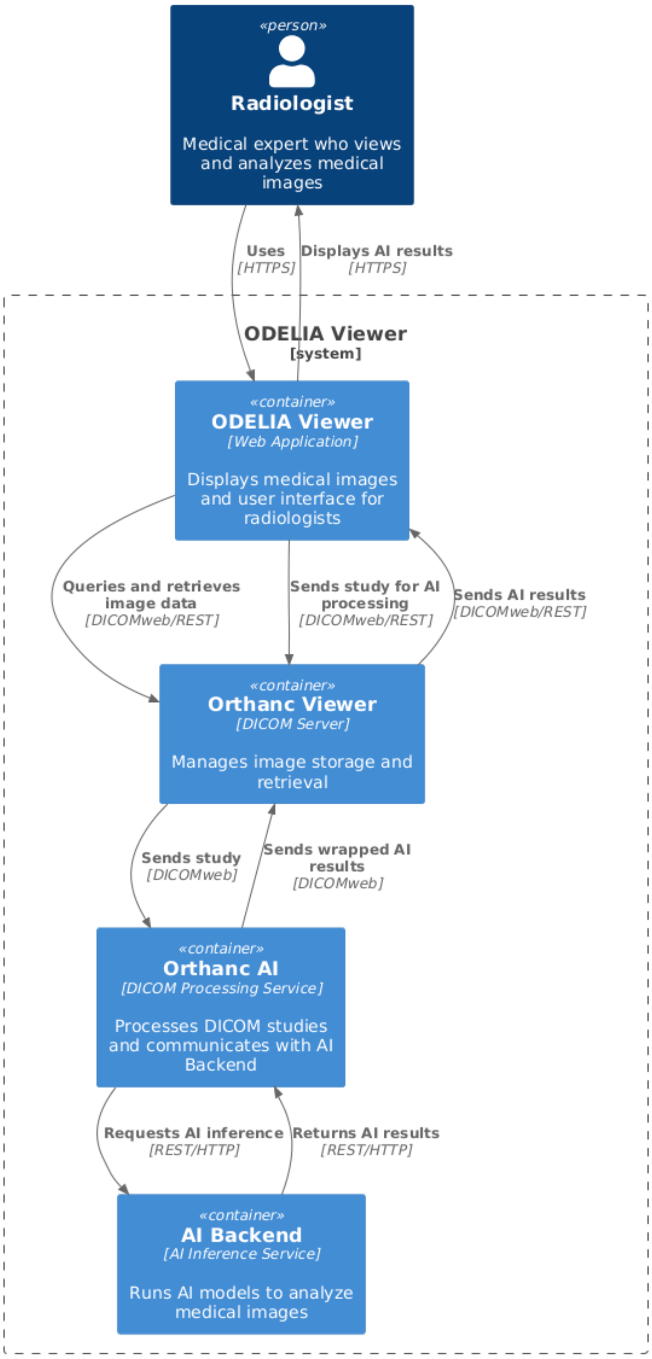

Figure 1: Container Diagram outlining the key systems of the ODELIA Viewer and their interactions

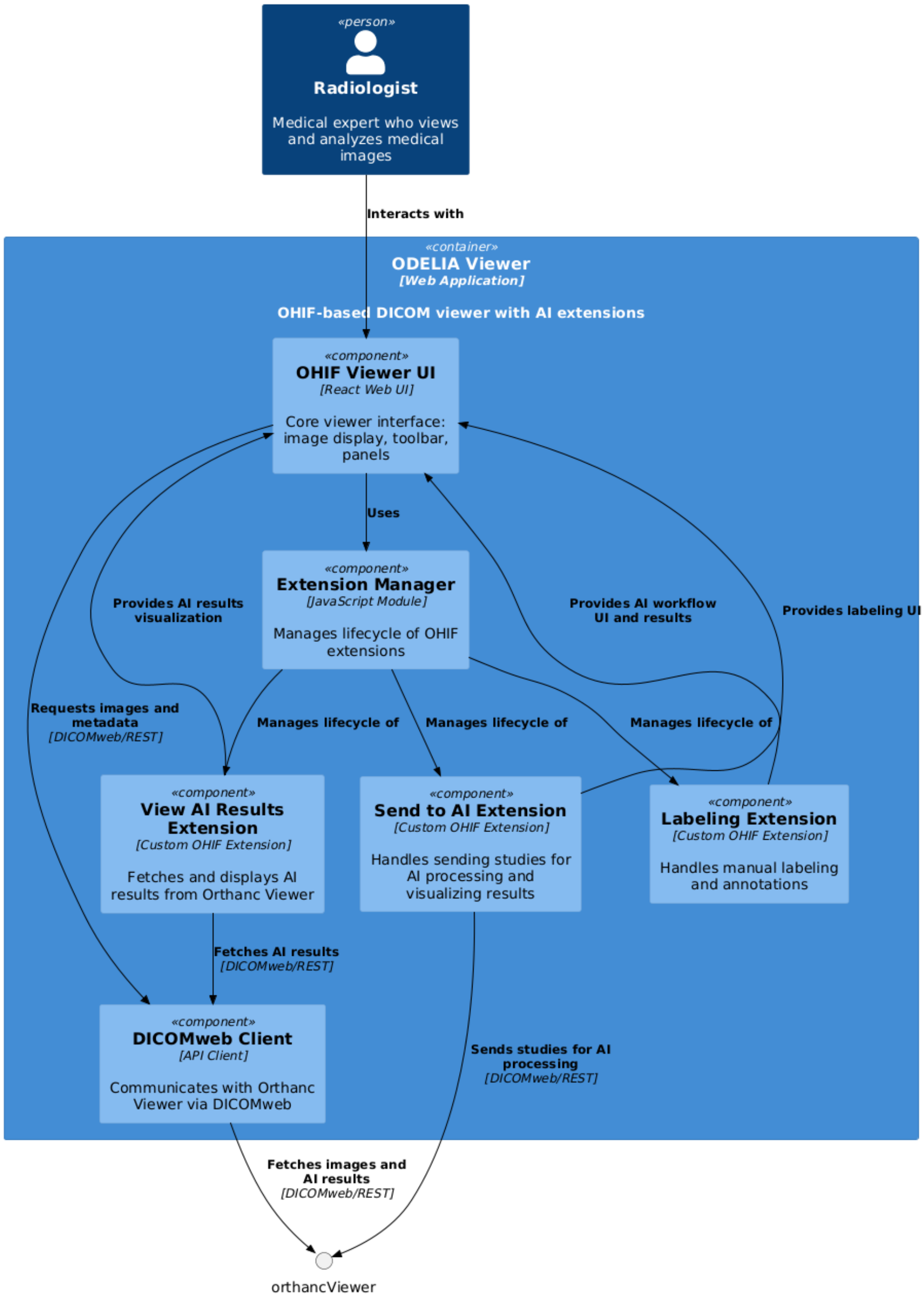

Figure 2: Component diagram: Further breakdown the ODELIA Viewer container into internal modules and extensions.

Development methodology

For the development process, we adopted agile methodologies commonly used in software engineering. JIRA is utilized as a product and task management tool, with a Kanban board to track active development tasks. For source version control (SVC), we use a combination of Git and a Bitbucket repository. To promote customer collaboration, we hold weekly meetings to discuss progress and set priorities for upcoming development tasks. We regularly demo iterations of our work to de-risk software development and ensure that the solution remains aligned with customer needs.

SECTION 3: WORKFLOW

This section provides a workflow diagram (Figure 3) which shows how the radiologist interacts with the ODELIA Viewer, and how the viewer triggers further components of our system to retrieve MRI examinations, trigger inference by the AI model, and display results.

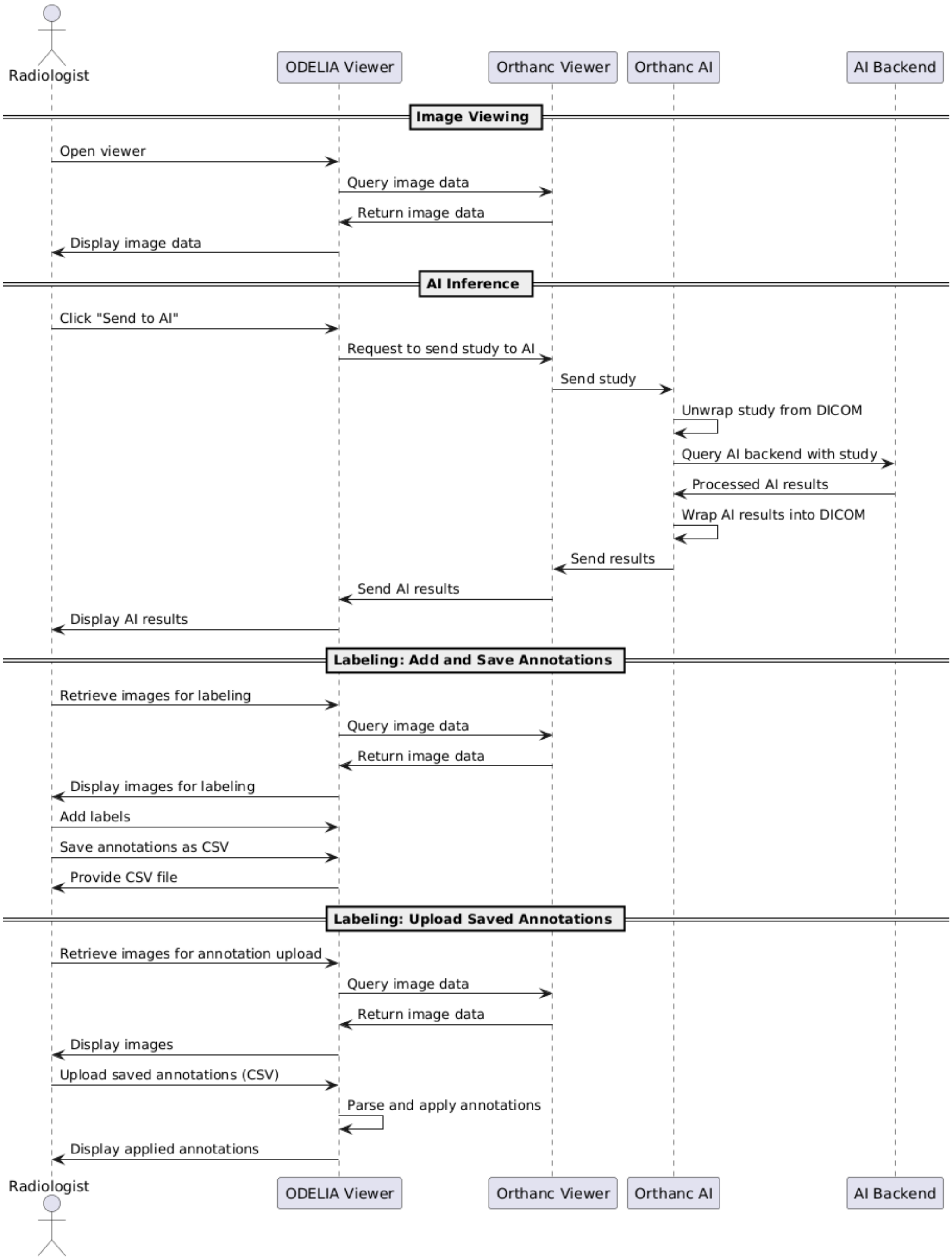

Figure 3: Workflow diagram of the ODELIA Viewer

SECTION 4: LAUNCH ODELIA WEB VIEWER

After the launch of the ODELIA Viewer, the radiologist will be presented with a list of MRI examinations. The radiologist can click on any examination (Figure 4), and open it in the ODELIA Viewer (Figure 5).

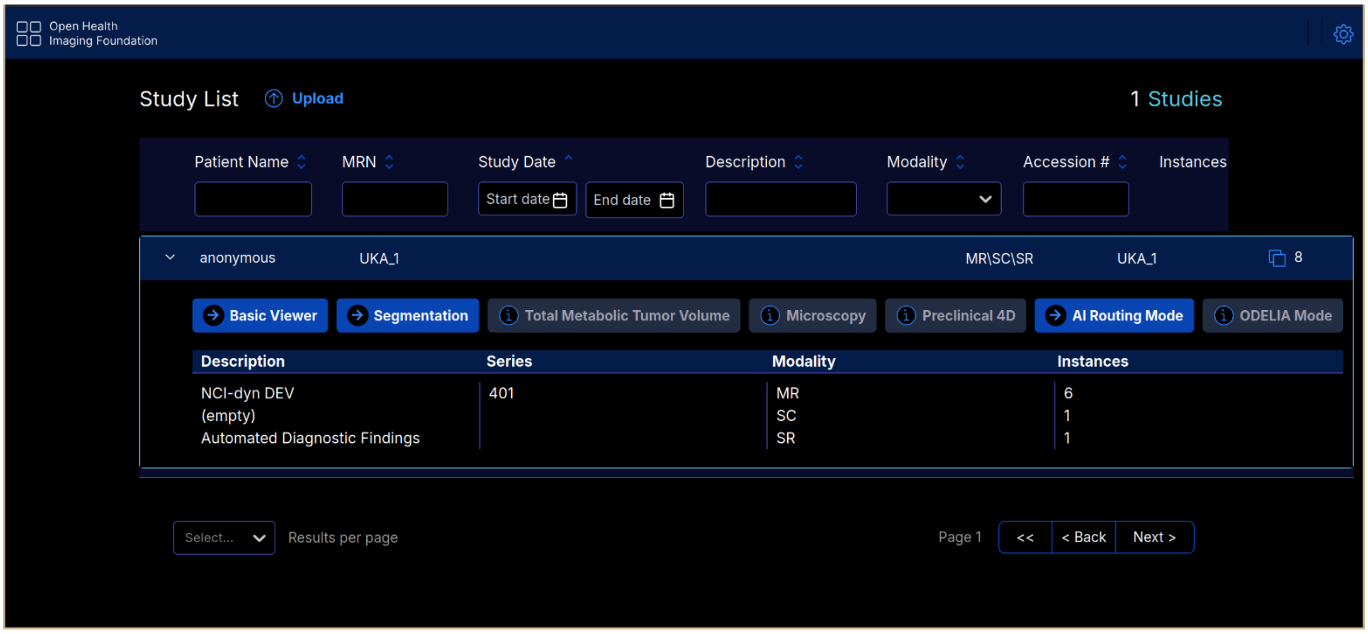

Figure 4: Functionality to open an MRI examination in the ODELIA Viewer.

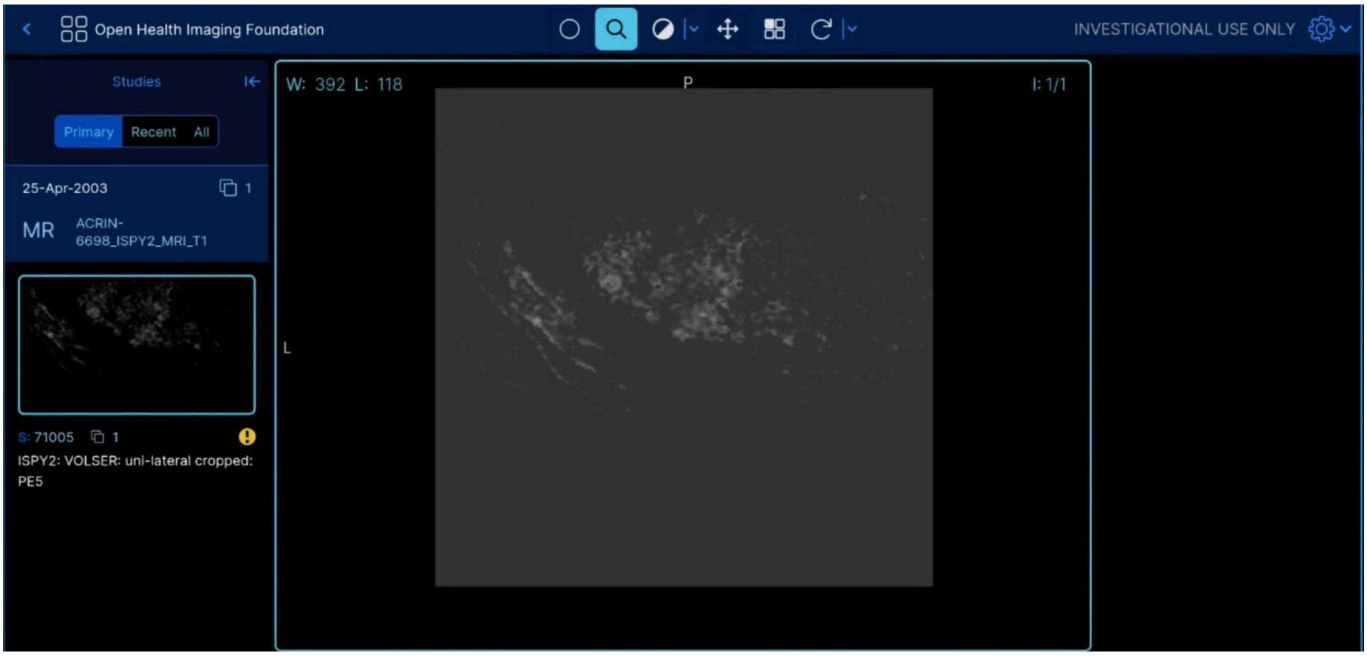

Figure 5: In the center of the viewer, the selected MRI study is displayed. The radiologist can scroll through the individual slices of the MRI scan and use the image enhancement tools at the top of the viewer to adjust the visualization to their specific needs. On the left and right sides of the main view, panels provide additional functionality, as described in the following sections.

SECTION 5: LABELING FUNCTIONALITY

A convenient labelling functionality is essential to efficiently generate the manual annotations required for AI model training. Figure 6 depicts the labeling functionality we have implemented in the ODELIA Viewer.

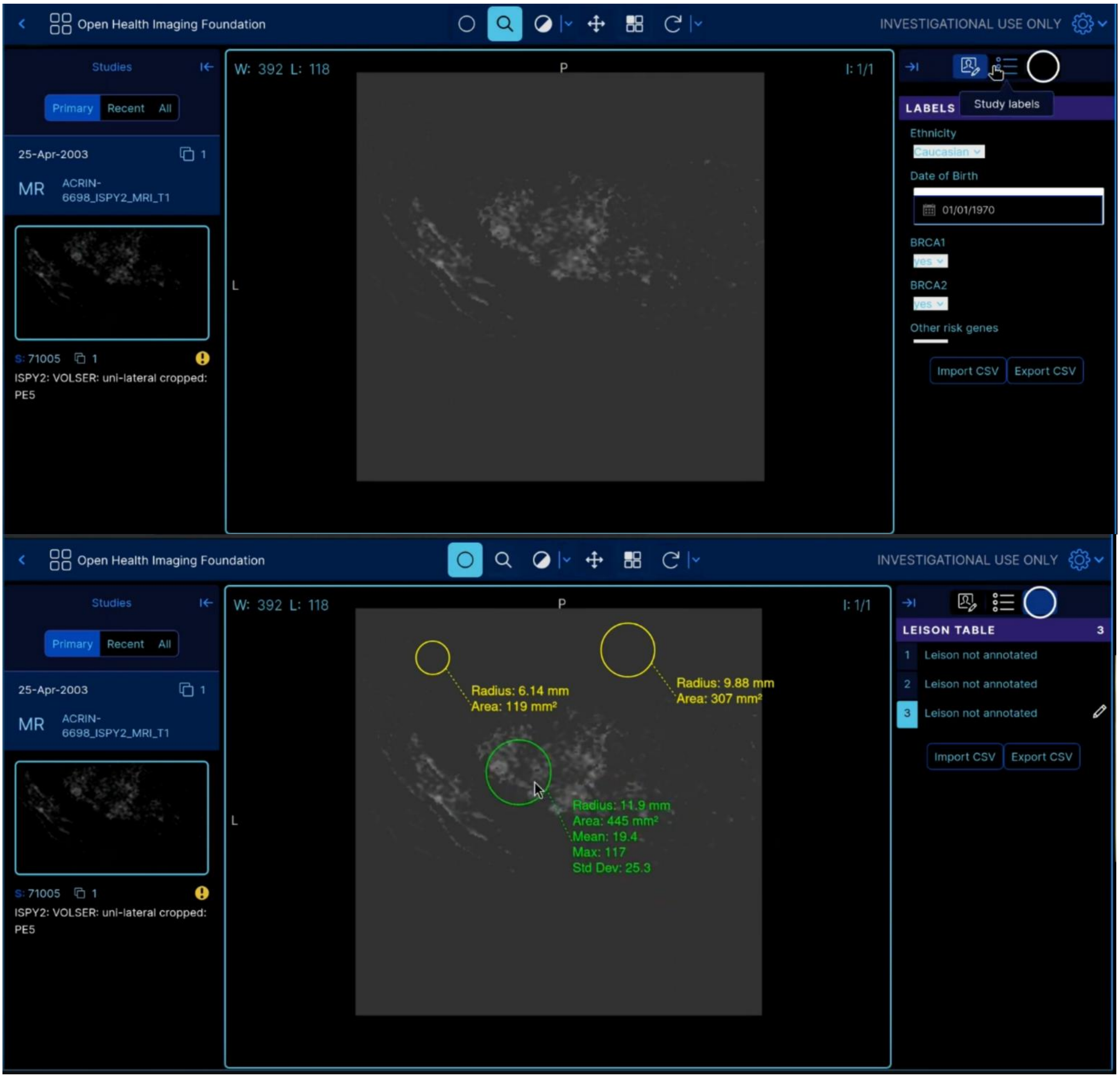

Figure 6: Labeling functionality of the ODELIA Viewer. On the right side, the radiologist has access to a labeling panel, where one can provide labels for the patient, the current MRI scan (study-labels, upper screenshot) and multiple lesion annotations within each scan (lesion table, lower screenshot).

From the technical side, the labeling extension we have developed is a reusable functionality package, which provides an ability to provide three types of labels, namely Patient level labels, Study level labels and individual lesion level labels.

Patient and Study labels are implemented in the ODELIALabel component, which extends OHIF measurements and provides key-value storage of labels for corresponding levels. The labels themselves are a set of configurable options. After discussion with stakeholders we identified three types of labels, namely Single choice (such as Ethnicity), Date (such as patient's date of birth) and Numerical (such as BIRADS score).

Lesion labels required a different approach, because there could be multiple lesions per study. We implemented a custom Cornerstone-based tool, which stores both the label (in the previously described format) and the location, which is defined as a CircleROI tool from the cornerstone library. Labeling extension is also responsible for importing and exporting labels to facilitate the data collection process.

SECTION 6: SEND TO AI FUNCTIONALITY

This section provides an overview of *Send to AI functionality* for ODELIA Viewer. From the radiologists point of view, we have implemented an additional panel at the right side of the screen as OHIF-based extension (Figure 7). Within this panel, the radiologist can push the '*Route to AI button'* to trigger AI inference, or update the models (Figure 8).

Figure 7: Example of the AI Routing panel (right side of the screen)

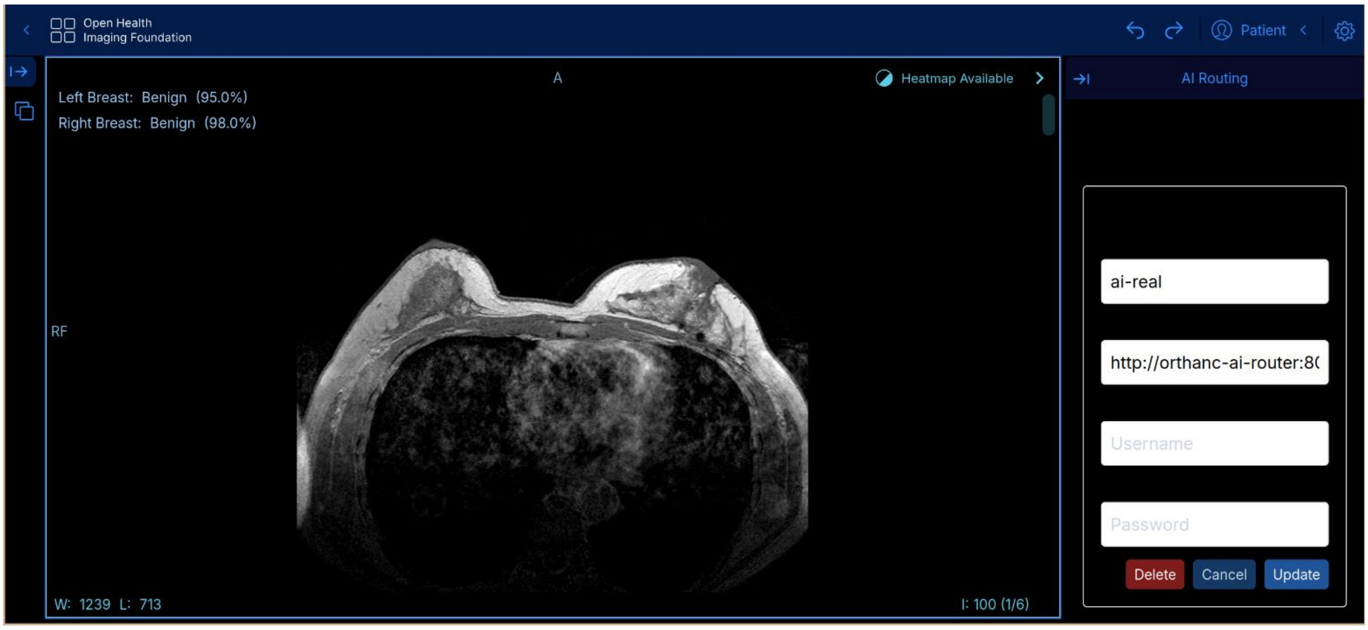

Figure 8: Functionality for modifying AI Models. The user can either add a new model by entering the corresponding web address as provided by the AI engineer, or delete a model. Once the user clicks update, the model information is updated in the system and the user is brought back to the previous view (Figure 7).

Architecture and design choices

As the ODELIA consortium consists of multiple consortium members, all with their different IT infrastructure, requirements, and scanning sequences, the ODELIA Viewer needs to be able to deal with a variety of models and input data.

To facilitate the use of different models, model outputs and general reusability we opted to follow IHE AI Workflow for Imaging (AIW-I) technical framework. This framework was published by Integrating the Healthcare Enterprise organizations and outlines workflow use cases involving the request, management, and performance of AI Inference on digital image data acquired by an imaging modality. By making the ODELIA Viewer and its adjacent components IHE AIWI-I compliant, we hope to broaden the impact and adoptions of this technology into pre-clinical and clinical practice.

The IHE AI Workflow for Imaging (AIW-I) framework defines standardized patterns for managing AI inference in imaging environments. It formalizes the roles and interactions required to request, assign, execute, and complete inference tasks in a vendor-neutral, interoperable manner using DICOM Unified Procedure Step (UPS) objects.

ODELIA Viewer adopts the Push Workflow model described in Section 50.4.2.3 of the AIW-I technical supplement. Compared to the other two AIW-I workflow types (Subscription and Polling), the Push Workflow offers a more centralized and deterministic approach to task orchestration. This suits the ODELIA deployment model, which integrates a viewer frontend with dynamic AI task routing across distributed inference services.

In the Push Workflow, the interaction is structured around four primary roles:

1. Task Requestor: Creates a UPS workitem describing the AI inference task, including input references (e.g., StudyInstanceUIDs), task metadata (e.g., procedure codes), and desired output storage location.
2. Task Manager: Receives the original workitem and assumes responsibility for selecting the appropriate inference engine. It pushes a copy of the workitem to a specific Task Performer, and monitors task progress.
3. Task Performer: Receives the workitem, fetches the necessary input data, runs inference, stores the results, and updates the Task Manager on status and completion.
4. Image Manager: Manages storage and retrieval of input and output DICOM objects (e.g., MR images and Structured Reports).

The ODELIA system, illustrated in the C3 diagram (Figure 1, implements the AIW-I Push Workflow roles using the following components:

- Orthanc Viewer: This component fulfills three roles in the Push Workflow model:
  - As Task Requestor, it creates the request workitem when a user triggers an AI inference from the ODELIA Viewer interface. The workitem includes references to input studies, desired inference task type, and output destinations.
  - As Task Manager, it selects an appropriate Task Performer (i.e., a specific Orthanc AI instance) based on task metadata and available models, then pushes a copy of the request and necessary study data to that performer. It also monitors the task lifecycle via status updates.
  - As part of the Image Manager, it serves both the input studies and AI-generated results to the ODELIA Viewer using DICOMweb.
- Orthanc AI (model-specific instances): These components act as Task Performers. Each one is tailored to a particular AI model or task. Upon receiving a workitem pushed by Orthanc Viewer, the instance retrieves input data from Orthanc Viewer, executes the inference, stores results (e.g., DICOM SEG, SR), and updates the status back to the Task Manager.
- ODELIA Viewer (Frontend): The user-facing interface used by clinicians and researchers. It does not directly interact with AI models. Instead, it visualizes studies and results via DICOMweb served by Orthanc Viewer, and initiates inference indirectly through viewer-side actions that trigger UPS creation.

This design enables dynamic orchestration of heterogeneous AI models without embedding model-specific logic into the viewer. It also aligns with IHE standards, facilitating interoperability and regulatory compliance in future deployments.

Whiles we could successfully solve the challenge of variation in AI model, we still need to account for the variety of input data. In clinical routine, different MRI sequences are used depending on the clinical question, the individual situation of the patient, and the technical capabilities of the MRI scanner. Ideally, we could extract sequencing information from the DICOM labels of the examinations and implement an automated solution, however a survey among ODELIA consortium members and expert interviews revealed a lack of standardization MRI sequences. While we're still exploring this solution as part of our Product Management Discovery process, the most likely implementation will be to provide the radiologist with an option to select the appropriate sequences manually. The current solution is optimized for the data structure of UK Aachen

SECTION 7: DISPLAY AI RESULTS

This section describes *Display AI Results* functionality, which provides an ODELIA specific functionality to display the results of AI models developed by the consortium. A customized panel on the left side displays all AI results that have been processed (Figure 9). By clicking on any item in this panel, the corresponding results will be shown in the upper left corner of the main view. In case the model provides a heatmap, e.g. for highlighting the areas of importance for the model decision, the radiologist can click on the heatmap button in the upper right corner. The heatmap will be shown in a side-to-side comparison.

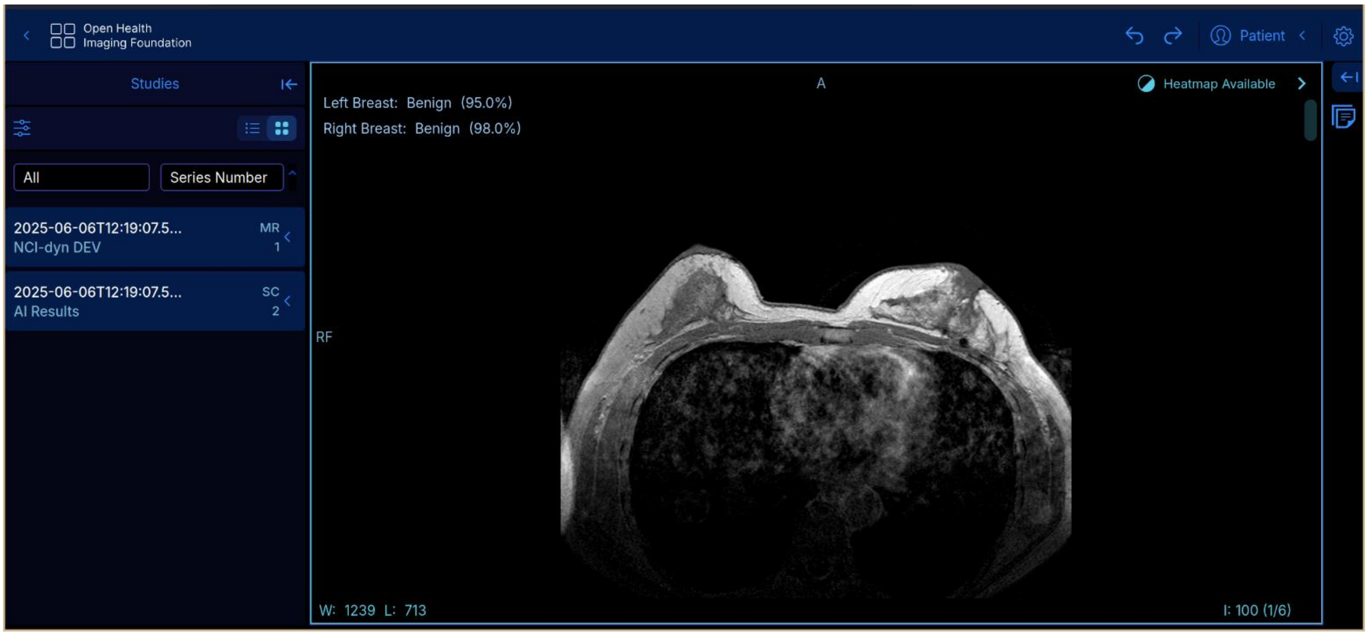

Figure 9: Display of AI results. The panel on the left side contains both the original study and the AI results. A text in the upper left corner of the main view presents the model classification and confidence level for each breast individually.

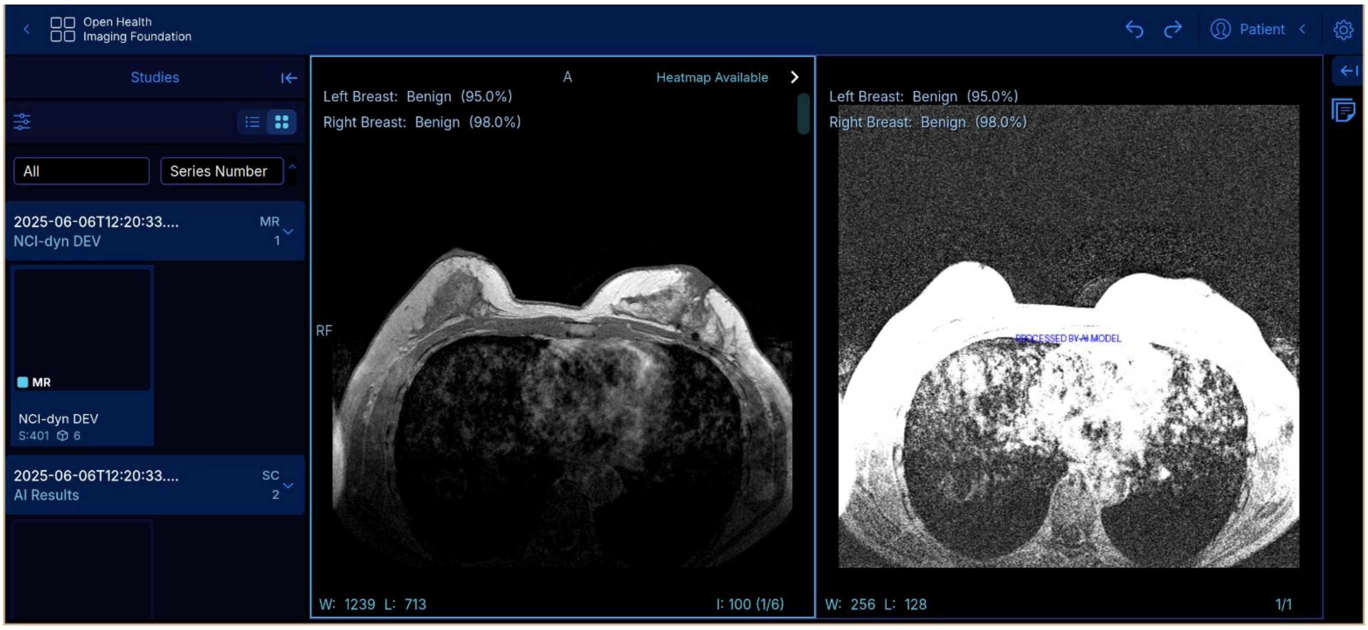

Figure 10: Display of heatmap. When clicking on the "Heatmap Available" button at the top right corner of the image, a side-by-side view displaying a model heatmap will be displayed. Bright areas indicate areas of high attention of the AI model.

From a technical perspective, we have implemented this functionality as OHIF extension to provide several key components to the radiologist:

1. AI study panel: A rework of OHIF study panel with specific UI tweaks for AI results. Namely, it groups AI results based on DICOM tags and displays model name and AI analysis date.
2. Comparison viewport: A custom Viewport component, that is used for classification result display and allows for heatmap comparison, if heatmap is provided as DICOM SC.
3. AI result service: This service, like the annotation service in OHIF, detects AI results from raw DICOM objects, implements hydration and AI result tracking across the extension.

CONCLUSION

StratifAI successfully delivered a customized MRI viewer, enabling consortium partners to test AI models directly through our tool. This achievement marks a major milestone in supporting the transition of the Swarm Learning AI models trained in the ODELIA project.

Looking ahead, the next key priority is to implement feedback functionality that will enable the structured collection of feedback from the radiologist on the AI model performance. Another focus will be to implement a strategy to deal with the variation in scanning sequences (see Section 6: Send to AI Functionality).

With the viewer now in the hands of consortium members, user feedback is actively being gathered and will be incorporated into the final deliverable. Thanks to clear prioritization and strong collaboration with other ODELIA consortium members, the project is back on track, and we anticipate delivering Deliverable D4.3 on time.