Ref. Ares(2025)10768456 - 05/12/2025

openZDM

OPEN PLATFORM FOR REALIZING ZERO DEFECTS IN CYBER PHYSICAL MANUFACTURING

D4.5 – Integrated platform and apps – final version

openZDM

Version	1.0
WP	4
Delivery Date	5/12/2025
Dissemination level	PU
Deliverable lead	INTRA
Authors	LMS, INTRA, MSI, IPB, TECNALIA, HABBER, UPORTO
Reviewers	LMS, MSI, TECNALIA
Abstract	This document is a demonstrator of the openZDM platfom and its applications. The document presents the basic concepts related to the openZDM platform, the platform's configuration aspects, and how the platform operates including its applications such as the Digital Twin Toolset, the Quality Assessment Modules, and the Decision Support Tool.
Keywords	Platform, Digital Twin Toolset, the Quality Assessment Modules, and the Decision Support Tool, Asset Administration Shell.
License	This work is licensed under a Creative Commons Attribution-No Derivatives 4.0 International License (CC BY-ND 4.0). See: https://creativecommons.org/licenses/by-nd/4.0/

Dissemination Level:	
PU	**Public, fully open**
SEN	Sensitive, limited under the conditions of the Grant Agreement
Classified R-UE/EU-R	EU RESTRICTED under the Commission Decision No2015/444
Classified C-UE/EU-C	EU CONFIDENTIAL under the Commission Decision No2015/444
Classified S-UE/EU-S	EU SECRET under the Commission Decision No2015/444
Type	
R	Document, report (excluding the periodic and final reports)
DEM	**Demonstrator, pilot, prototype, plan designs**
DEC	Websites, patents filing, press & media actions, videos, etc.
DATA	Data sets, microdata, etc.
DMP	Data management plan
ETHICS	Deliverables related to ethics issues.
SECURITY	Deliverables related to security issues
OTHER	Software, technical diagram, algorithms, models, etc.

Funded by the European Union

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or EUROPEAN HEALTH AND DIGITAL EXECUTIVE AGENCY (HADEA). Neither the European Union nor the granting authority can be held responsible for them.

Version History

Version	Date	Owner	Author(s)	Changes to previous version
0.1	02-09-2025	INTRA	INTRA	Outline
0.2	04-09-2025	INTRA	All WP4 partners	Working document
0.8	21-11-2025	INTRA	All WP4 partners	Full draft ready for internal review
0.9	2/12/2025	INTRA	All WP4 partners	Internal review
1.0	5/12/2025	INTRA	All WP4 partners	Final draft

Table of Contents

List of Abbreviations & Acronyms

AAS	:	Asset Administration Shell
CSV	:	Comma-Separated Values
DAT	:	Data-Driven Quality Assessment Module
DL	:	Deep Learning
DST	:	Decision Support Tool
DTT	:	Digital Twin Toolset
HTTPS	:	Hypertext Transfer Protocol Secure
ISO	:	International Organization for Standardization
LCA	:	Life Cycle Assessment
LCID	:	Life Cycle Inventory Data
LSTM	:	Long Short-Term Memory
ML	:	Machine Learning
MTBF	:	Mean Time Between Faults
MQTT	:	Message Queuing Telemetry Transport
NDI	:	Non-destructive inspection
OWASP	:	Open Worldwide Application Security Project
RFR	:	Random Forest Regressor
SHAP	:	SHapley Additive exPlanations
SoA	:	Service-oriented Architecture
SVR	:	Support Vector Regression
WP	:	Work Package
ZIP	:	Compressed Archive Format
ZDM	:	Zero-Defect Manufacturing
ZSL	:	Zero-Shot Learning

List of Figures

List of Tables

Executive Summary

This document serves as a demonstrator for the openZDM Integrated platform and its applications. The document provides an overview on how the software components developed within the openZDM project, along with "off-the-shelf" components are used to deliver the platform's functionalities to its users. The document is structured in five sections as follows:

Section 1 provides an overview and explanations of the technologies adopted or developed in openZDM along with how they are applied in the project. More specifically, the role of Asset Administration Shells, Authentication & Authorization, Digital Twin Toolset, Quality Assessment Modules, Decision Support Tool, and the openZDM platform core are presented so that the reader may understand their application in the platform.

Section 2 explains how the integration approach covering the UI integration based on IFRAMES and the backend integration using Keycloak, Eureka, provision of API documentation, service monitoring, and message exchange with MQTT.

Section 3 explains how the various components are configured so that the final users of the platform can access the platform's functionalities for non-destructive testing data visualization, process monitoring, defect prediction, and decision support. In this section a set of screenshots and explanations are provided for the configuration of the platform core (i.e. user management, application configuration, notifications, data export), the Digital Twin Toolset, the Configurator application, the Quality Assessment Modules, and the Decision Support tool.

Section 4 demonstrates the functionalities offered by the platform to its users after all appropriate configurations have been carried out. It covers the main user interface, and the openZDM applications: Digital Twin Toolset, Quality Assessment Modules, Decision Support Tool. The Quality Assessment Modules section covers all 14 tools developed in the project. The last part of this section covers AAS type 3 functionalities and provides links to videos.

Section 5 discusses the security provisions of the platform, scalability, accessibility, and replicability. Moreover, the following performance metrics have been validated:

- Support for cloud and hybrid deployment
- Support for multiple users with different workflows, preserving data isolation.
- Data turnover time less than 4.9 seconds.
- Digital twins to be successfully used in multiple use cases.
- Ratio of false overall quality assessment evaluation of data-driven analytics.
- Digital twins update with live data from at least 3 sources.
- At least one data-driven tool has been developed and validated in each use case.

Section 6 details the conclusions of this document and the development process in openZDM. A brief overview of the conclusions is as follows:

- openZDM applications are complex, often consisting of multiple components and configurations; some can operate standalone or in combination.
- Despite efforts to simplify user interfaces, complexity remains—especially in tools like the Digital Twin Toolset, which supports data-driven, physics-based, and hybrid modelling approaches requiring advanced domain knowledge.
- To further enhance accessibility the platform allows for configuration of views per user meaning that specific users may have a reduced amount of information whereas others may have more complex views. Moreover, multi-language support was added to support users that are not native English speakers. Lastly, an inclusive design approach was used during the development of the openZDM platform.
- The platform relies heavily on Asset Administration Shells (AAS), a still-maturing concept with limited implementations and documentation, leading to workarounds (e.g., direct database access, lack of AAS search APIs, no native support for historical data storage).
- Development of Quality Assessment Modules faced challenges due to heterogeneous legacy systems and data collection complexities; the openZDM AAS Middleware helped standardize data access.
- The AAS proved valuable for integrating sensor equipment (e.g., NDIs) by abstracting hardware/software via the AAS model and API.
- For broader industrial adoption, improvements are needed: more robust AAS implementations for scalability, and simplified, more user-friendly interfaces for tools like Digital Twin Toolset and Decision Support Tool to make them accessible beyond expert engineers.

Introduction

This document serves as a demonstrator for the openZDM Integrated platform and its applications. The document provides an overview on how the software components developed within the openZDM project, along with "off-the-shelf" components are used to deliver the platform's functionalities to its users. The document is structured in five sections as follows:

- Section 1 provides an overview and explanations of the technologies adopted or developed in openZDM along with how they are applied in the project.
- Section 2 explains how the various components developed in the project were integrated together.
- Section 3 explains how the various components are configured so that the final users of the platform can access the platform's functionalities for non-destructive testing data visualization, process monitoring, defect prediction, and decision support. In this section a variety of screen shots from the project's prototypes is included.
- Section 4 shows the functionalities offered by the platform to its users after all appropriate configurations have been carried out. It covers the main user interface, and the openZDM applications (i.e. Digital Twin Toolset, Quality Assessment Modules, Decision Support Tool).
- Section 5 details the conclusions of this document and the development process in openZDM.

1 Essential concepts

This section introduces the various concepts that are associated with the openZDM platform and its applications. It provides explanations on:

1. OpenZDM architecture
2. Asset Administration Shell
3. Authentication and authorisation
4. Digital Twin Toolset
5. Quality Assessment modules
6. Decision support tool
7. openZDM platform
8. Configurator

1.1 openZDM architecture

The openZDM platform is a group of software components working together to support production networks' zero-defect processes. The platform accesses data from a variety of data sources, including NDIs, equipment deployed at a manufacturing company's shop floor, and other legacy systems (such as databases). The data are stored and processed in the platform by a variety of tools having as the following key objectives:

1. Monitoring of processes to identify deviations that cause defects, through the application of Digital Twins.
2. Data analysis for quality assessment and prediction of defects.
3. Decision support through the evaluation of alternative zero-defect manufacturing strategies and the suggestion of adaptation strategies.

The openZDM architecture that facilitates the goals of the project is illustrated in Figure 1. The definition of this architecture is based on the Reference Architectural Model Industrie 4.0 (RAMI 4.0) [1], and the work carried out in the context of Sense&React [2], and Boost 4.0 [3] architectures.

Figure 1: The openZDM platform architecture [4]

The architecture of the openZDM platform comprises three tiers. Each tier can be either deployed on the cloud, locally deployed or hybrid. Each layer within a tier of the architecture is responsible for grouping together specific components. The platform layers include the following horizontal layers:

- **External data source and infrastructural layer:** This layer represents all the assets of the end-users that need to be connected to the openZDM platform, such as NDIs, existing equipment, and legacy systems.
- **Integration layer:** The integration layer facilitates the communication between the platform and all the assets. In this layer, the openZDM AAS Middleware is applied to represent assets as AASs and manage the integration of NDIs.
- **Data layer:** The data layer's role is to intercept messages from all the assets, store data, and provide access to data.
- **Service layer:** In the service layer, components that implement some of the platform's functionalities are included (e.g. notifications backend service, service discovery module, platform configurator backend), and the layer facilitates access to the data layer from applications in the Business layer (Service API).
- **Business layer:** This layer groups together the functionality that is offered to the users and supports their business objectives. At a minimum, this layer can be conceived as a User Interface that provides access to the data of the platform. In more complex cases, the User Interface is the entry point for openZDM applications offering production monitoring, data analysis, and decision support.
- The architecture also includes the three vertical layers, which provide functionalities to all the horizontal layers:
- **Authentication & authorisation:** Provides functionalities for user and services identification, mechanisms to secure communications and authorisation mechanisms that permit access to resources of the platform. This layer is realised by the Identity Server of the platform that relies on the OAuth2.0 protocol [8] for authorisation.
- **Logging & Monitoring:** This layer logs user activity and records system errors.
- **Data sharing:** The data sharing layer enables the integration of the platform into a data sharing ecosystem.
- Sections 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, and 1.8 provide further details on the most important components/technologies applied in the various layers. For the ASS Middleware, several details have been provided in Deliverable "D3.6 – NDIs SW apps integration to openZDM platform", and thus only an update is provided to complement the content in that deliverable.

1.2 Asset Administration Shell (AAS)

OpenZDM uses the concept of AAS to facilitate the integration of assets into the platform. AAS is the digital representation of an asset and consists of several sub models in which all the information and functionalities of a given asset, including its features, characteristics, properties, statuses, parameters, measurement data and capabilities, can be described [5],[6]. In the openZDM platform, assets are equipment or processes such as a furnace, NDIs developed in the project, and products. Deliverable "D3.6 – NDIs SW apps integration to openZDM platform" reports in detail the process of modelling an NDI and integrating it into the platform in section 2.2. In addition, it reports on the openZDM AAS Middleware, which is a piece of software that facilitates interaction with the AAS model. The difference between NDI AASs and Process/Products is only the AAS model; these are reported in section 1.2.1 of this document.

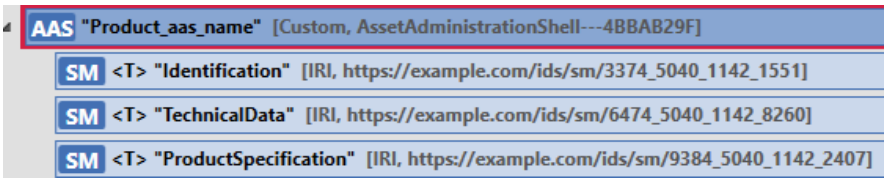


Figure 3: Product AAS template

Figure 2: Process ASS template

1.2.1 Modelling a Process/Product using AAS in openZDM

The information corresponding to the products of the different use cases can be grouped into three sections: Identification submodel, **TechnicalData** submodel, and **ProductSpecification** submodel. The first two submodels can be explained in the same way as for the NDI. While the **ProductSpecification** submodel represents all the product information collected during the manufacturing process (From resource operational data). Figure 2 depicts the AAS template for Products.

The modelling of the processes was done in a similar way for the NDIs. The ***TechnicalData*** submodel was not modelled because, in many cases, it was not possible to get the technical information from the equipment. Therefore, for the sake of consistency, the **TechnicalData** submodel was not represented. Figure 3 depicts the AAS template for processes. A standardised version of the **TechnicalData, Capabilities**, and **HierarchicalStructures** submodels from IDTA [7] was reused in the project.

1.3 Authentication and authorisation in openZDM platform

Authentication identifies the platform users and requires a username and password. Authorisation is role-based and defines what can be accessed by each user. **Keycloak** is an open-source tool that handles both authentication and authorisation for openZDM applications. It supports **OAuth2** [8], which is a standard way to let users log in and give permissions securely. A step-by-step process for authentication and authorisation is as follows:

1. User tries to access the user interface
   - If no prior login details are stored in the browser, the user is redirected to Keycloak's login page.
2. Keycloak authenticates
   - The user enters their username and password.
   - Keycloak checks their credentials and, if correct, confirms their identity.
3. Keycloak issues tokens
   - After login, Keycloak gives the openZDM user interface the following tokens:
     - Access Token: proves the user is allowed to use certain parts of the app.
     - Refresh Token: lets the app get a new access token when the old one expires.
     - ID Token: contains info about the user (like name and email).

4. The various parts of the user interface use the tokens
   - To access backend services.
   - Adapt themselves to the user's role.
   - Backend services determine the user's profile.

In addition, the user interfaces provided by Keycloak are used to configure users and various other aspects of authentication and authorisation.

1.4 Digital Twin Toolset

The Digital Twin Toolset (DTT) allows the creation of new digital twins as well as the loading and visualisation of previously created ones. Specific steps have been defined for a user to undertake the creation of a new digital twin and include:

- Initialisation & asset definition: The first step in the process allows users to define a unique name for the digital twin being created. In addition, through this step, the use case the digital twin refers to is selected together with the accompanying assets that will be included in the digital twin, while the user also defines the defects that can be observed in the physical system.
- Modelling: In the modelling step, the core element of the digital twin is defined, its simulation model. The model can be based on data-driven or physics-based techniques, with a range of available ML, DL and physics-informed AI algorithms being available for user selection based on their needs for accuracy, explainability and ease of use.
- Workflow: In the workflow step, the user of the DTT is exposed to a workflow creation engine that allows the integration of the created model with real-world or synthetic data. In addition, the workflow step enables DTT users to integrate model outputs with structured and unstructured databases, the AAS Middleware of the openZDM platform, and available Quality Assessment Modules.
- Visualization: The final step is the definition of the visualization. Two visualization options are available, a 2D-based and a 3D-based. Asset models are loaded automatically into a 2D or 3D canvas where the user can position them. Lastly, the DTT allows the introduction of information panels embedded into the canvas to visualize asset data from the AAS Middleware of the platform.

1.5 Quality Assessment Modules

The Data Driven Quality Assessment Modules (DAT) in the openZDM platform provide advanced analytical capabilities, e.g., monitoring, diagnosis, prediction, and optimisation of manufacturing processes. Their purpose is to enable Zero-Defect Manufacturing (ZDM) by performing AI-based data analytics to enable monitoring and the early and real-time detection of process and product quality deviations and trends. In this context, this section offers a brief view of the DATs. The full technical details concerning their architecture, development, and integration are omitted as the focus of this document is on the demonstration of the developed tools.

The DATs were designed to be technology-neutral and use-case-independent, ensuring that they can be flexibly applied across different industrial environments. This design philosophy enables users to select and combine the most relevant tools from the openZDM catalogue to build customised quality-assessment solutions tailored to their specific manufacturing needs. Additionally, the DATs follow a Service-oriented Architecture (SoA) approach, in which the functionalities of each module are encapsulated as independent services that can be offered to and consumed by other components of the system. Within the openZDM platform, the interface between the DATs is implemented through mechanisms that allow seamless data exchange and integration with other components or NDIs.

The final openZDM release includes 14 DATs (labelled DAT#0 to DAT#13), grouped into four main functional clusters, namely:

- **Pre-processing:** encompasses tools for data cleaning and synthetic data generation.
- **Monitoring:** focuses on real-time visualisation and detection of anomalies and defects.
- **Prediction & Classification:** supports the early detection of defects and quality assessment.
- **Descriptive & Prescription:** provides for decision support through explanations of defect causes and recommending corrective actions.

For validation purposes, the DATs were instantiated and configured to address the specific requirements of the use cases defined in the project (i.e., VWAE, VDLWEW, SONAE, VIDRALA, and APTIV). In these implementations, each

DAT was adjusted according to the characteristics of the corresponding processes and technologies. In this regard, a brief description of each DAT, along with its classification according to the predefined functional clusters, is summarised in Table 1. Furthermore, while not all DATs are directly classified within the pre-processing cluster, the tasks associated with that cluster are common and essential steps shared across all DATs within the openZDM platform.

Table 1: Overview of the implemented Data Analytics Tools

DAT	Description	Cluster[1]
#0	Quality assessment and quality prediction analysis, e.g., straightness, thickness, and width of the trailing arms in real-time on-premises	B
#1	AI-based modules to identify anomalies and advice to operators to adjust control parameters	C
#2	A tool for managing the input and output data of each process and calculating the environmental impact of the product system	A
#3	Provide real-time defect prediction, explanation and optimal setpoint configuration, mitigating defect probability, and maximising performance	B and C
#4	Algorithm for simulating new products, helping to identify the probability of defects and provide insights to mitigate them, e.g., suggesting machine parameters	C
#5	Dashboards to follow-up the model's accuracy, comparing for the last production order the percentage of predicted defects with the real one	A
#6	Use IR images of thermal cameras and rejection thresholds to implement a prediction model for the horizontal distribution of the wall thickness and the quality of the bottles. The generalization of the segmentation service has been developed to create a model that predicts the mask of any bottle in a thermographic image. Both models are available	B
#7	Process reconfiguration strategy definition with dashboard visualisation for Gob features	A
#8	Prediction of problems in the rear-end and front-end gap and flush parameters, at the end of the body shop	B
#9	Prediction of the checking points (assembly line) that may fail according to data from previous stations (including the body shop)	B
#10	Analysis of the causes that impact the detected defects	C
#11	Real-time monitoring of the dimensional and gap, and flush measurements	A
#12	Monitoring and visualisation of critical quality-related KPIs, including MTBF, CoPQ, Cycle Time	A
#13	Reconfiguration of the battery cell charging process using cell-related data (voltage, internal resistance, capacity) to optimise cell recharging and avoid the generation of defective battery modules due to unbalanced cell charge	C

The development and deployment of these DATs are supported by an appropriate model training application (discussed in the following sections), which provides a unified interface for training AI models, evaluating their performance, and registering them on the platform. Once trained, DATs are containerised and orchestrated to ensure consistent deployment and integration across pilots.

Further details on the general deployment procedures and the operational functionalities of each implemented DAT are presented in the following sections of this document.

1.6 Decision Support Tool

The Decision Support Tool (DST) enables the creation and evaluation of alternative “what-if” scenarios against specific cost/benefit indicators. The tool is interconnected with the created digital twins by the DTT to tap into their digital twin models to perform feed-forward simulation of a manufacturing system. Simulation results are evaluated using specific optimisation components, providing its users with a time-step-based recommendation for a user-defined simulation time horizon.

Specific aspects of a manufacturing system can be optimised via the DST, defined via cost/benefit indicators. Such aspects can include the global warming potential of the system, in accordance with LCA standards (ISO

[1] A: Monitoring; B: Prediction & Classification; C: Descriptive & Prescription

14040[2]), the number of defects being produced, as well as use-case-specific indicators tailored for a manufacturing system's needs.

These functionalities enable the DST to provide its users with a set of values for process-related configuration parameters that can improve both the environment and the economic aspect of a system, based on user needs and the defined cost/benefit indicators.

1.7 openZDM platform core

The openZDM platform core of the main user interface, which is the main access point for all users, and integrates user interfaces from the Digital Twin Toolset, the Quality assessment modules, and the Decision support tool. In addition, the core consists of several backend services and components:

1. Keycloak for authentication and authorisation.
2. Eureka integration for service discovery.
3. Service monitoring using Prometheus.
4. Notification service to notify users of errors or other events through the user interface, emails, and SMS.
5. MQTT to facilitate communication using publish-subscribe.
6. The AAS Middleware.
7. Service orchestration based on Apache Airflow for triggering other parts of the platform (e.g. Quality Assessment Module execution) when required.
8. Services for storing user configurations for the user interface.
9. Databases for data storage.

The main purpose of the platform core is to provide the integration mechanisms required for the Digital Twin Toolset, the Quality assessment modules, and the Decision support tool, and in parallel, facilitate user interactions.

The various users of the platform are grouped by role and group. The role and group represent the user's access levels. Groups identify the user's pilot (e.g. one of the various companies using the platform), and they are used to separate data access between pilots. Roles represent the level of access within the pilot following roles are used:

1. Operator, user typically interested in the operation part of the platform. Each user with an operator role may have different configurations of applications that they view.
2. Administrator, a user with admin rights within a given pilot.
3. System administrator, a user with access to the whole system, including data from all pilots. Typically, this role is used to set up, supervise, and configure the platform.

1.8 Configurator

The Configurator is a low-code, canvas-based tool designed to define, parameterise, connect, and deploy Dockerised microservices as complete "apps." From its visual interface, it automatically generates the *docker-compose.yml* and related configuration files, enabling consistent and simplified deployment across different environments. Its architecture includes a frontend/backend pair for configuration and composition, a per-app "launcher" agent that retrieves definitions from the backend and manages service deployment on target devices, and monitoring services with a UI (using MQTT/MQTTS) to stream status, logs, and resource metrics.

Within the openZDM ecosystem, the Configurator is used to deploy and manage containerised applications, including Digital Twins, DAT-s and other software components. It integrates with core services such as Keycloak for authentication, Prometheus for monitoring, and the platform's notification service for logs. It supports both manual and automated deployment modes, allowing launchers to handle updates and orchestration across distributed setups, making it a central enabler for application deployment and lifecycle management in openZDM.

2 Integration approach

As evident from section 1.7, the openZDM platform requires integration on two levels:

1. The User interface level
2. The Backend service level

[2] https://www.iso.org/standard/37456.html

The first group deals with the connection of the individual user interfaces into the main user interface of the openZDM platform, and the second one deals with the integration of the various services.

2.1 User interface integration

User interface integration follows the IFRAME approach, which allows any kind of web-based user interface to be visualised through the main user interface of the platform.

The IFRAME approach enables the integration of user interfaces irrespective of the underlying development framework. All user interfaces are secured using Keycloak. For the user interfaces developed in openZDM the user profile and access token that is required to access any backend services are passed from the main user interface of the platform to the individual application user interfaces using the JavaScript window.postMessage() method, which is available on all browsers. This message contains the user's profile information (e.g. name, role, username) and the user's access token that can be used to access backend services.

2.2 Backend service integration

Backend service integration relies on the following aspects:

- Keycloak integration for authentication and authorisation.
- Eureka integration for service discovery.
- Provision of API descriptions using OpenAPI
- Service monitoring.
- Access to MQTT.

2.2.1 Keycloak integration

All APIs are accessed using JSON Web Token (JWT) issued by Keycloak. The tokens may identify users or clients (services). Tokens are included in requests to backend services using the appropriate headers.

2.2.2 Eureka integration for service discovery

The service discovery component is based on Spring Cloud Netflix - Eureka Server. Typically, integration is achieved using "off-the-shelf" libraries that enable Eureka integration through configuration. The configuration includes the ID to register in the Eureka Server, an SSL-enabled option, the port to be registered in the Eureka Server, the service URL of the Eureka Server, and the preferred IP address. Lastly, other applications can access the IP and port of each service using the various IDs registered in the Eureka Server.

2.2.3 Provision of API descriptions using OpenAPI

All services in openZDM provide their own API description using OpenAPI format. OpenAPI is a standard way to describe how a web API works and how to use it. It enables:

- Easy documentation because it automatically generates readable API docs from the source code.
- Interactive testing using tools like Swagger UI that let the developers try the API directly from a browser.
- Shared understanding for developers, testers, and clients who all see the same documentation.
- Automation, because it allows for generating client SDKs or server code automatically.
- Integration with security: because it allows for describing how OAuth2 or Keycloak authentication works right in the specification of the API.

2.2.4 Service monitoring

Service monitoring is achieved using Prometheus, an "off-the-shelf" open-source monitoring system. Prometheus uses predefined APIs to scrape logs from applications. The logs are stored in a type of database that can be later processed to determine component uptime and other relevant metrics.

2.2.5 Access to MQTT

For the cases where the publish/subscribe approach is required for communication between the various components, an MQTT Broker per pilot case is provided. The MQTT Broker is Mosquitto MQTT.

3 Platform configuration

The section provides a demonstration of the various configuration activities typically performed in the openZDM platform. It assumes that the System administrator role has already been configured. It also assumes that the required assets have been configured following the approach detailed in deliverable “D3.6 – NDIs SW apps integration to openZDM platform”.

3.1 Users, roles, and groups configuration

In this section, the system administrator user is used to create one role called operator, and one group called demo. Then the demo-user is created and assigned to the role and group. The main view when the system administrator is logged in is presented in Figure 4.

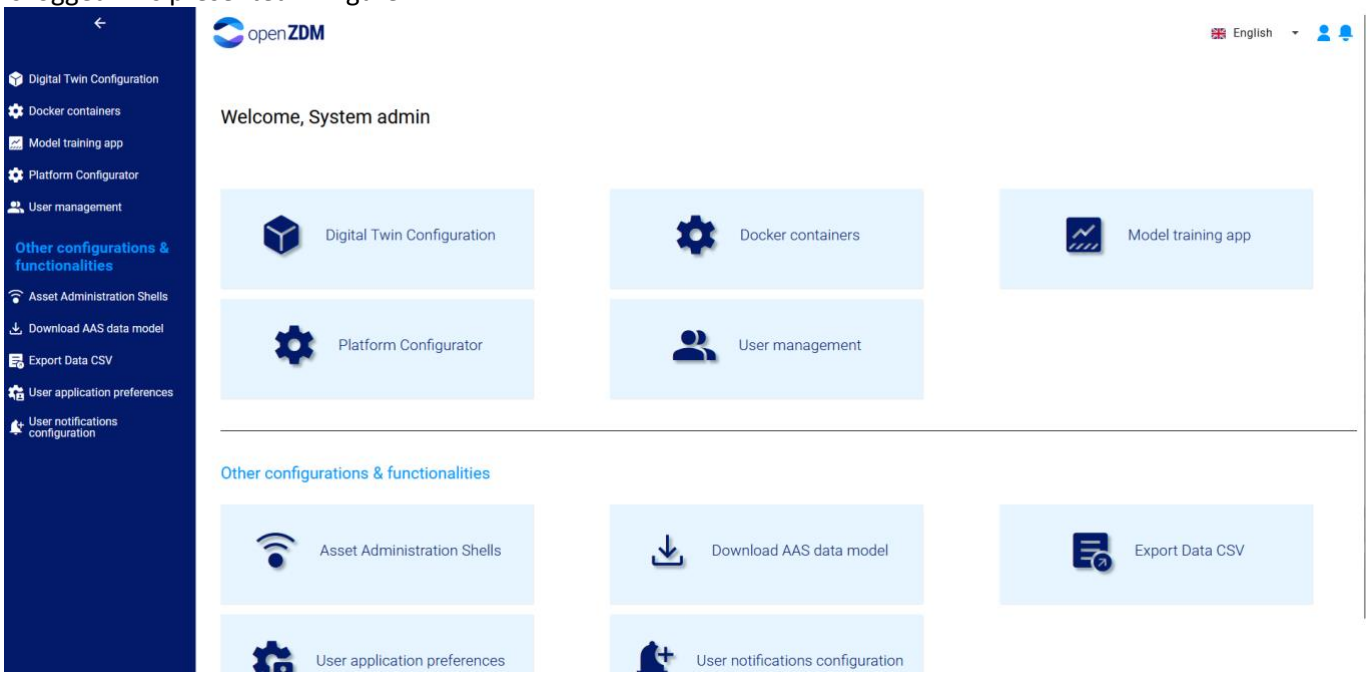


Figure 4: openZDM platform UI administrator main view

The button user management is used to lead the user into the Keycloak (see details in section 1.3) user interface. There, a new group, role, and user can be created. In Figure 5 the group is created.

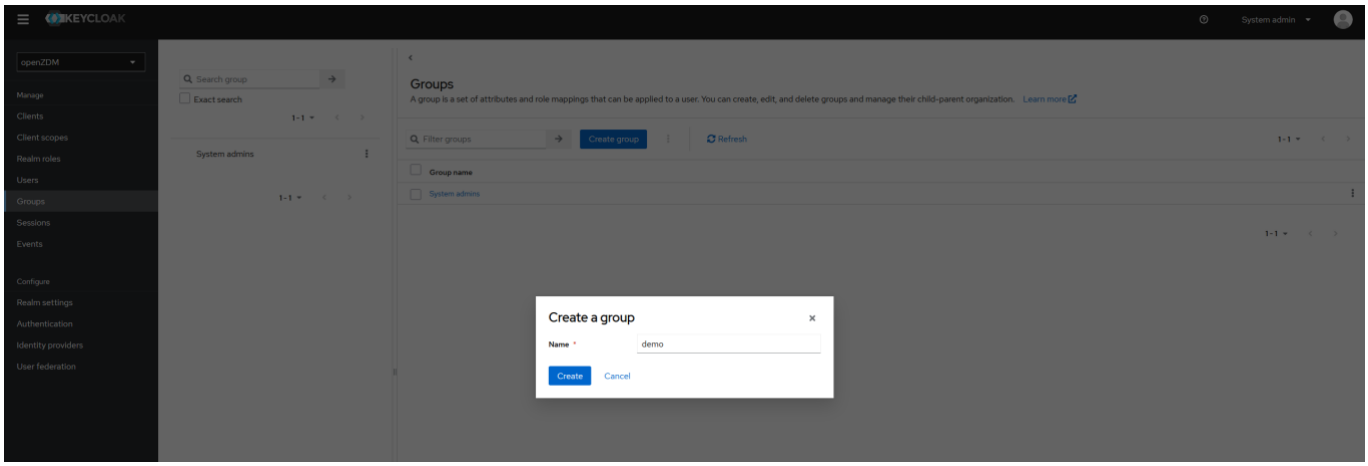


Figure 5: Keycloak group creation

In Figure 6 the operator role is created.

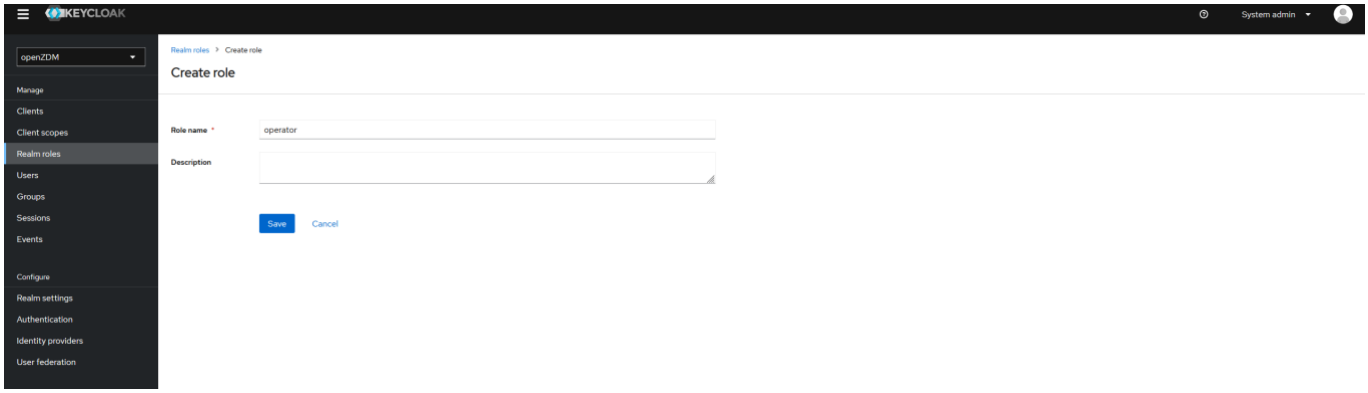


Figure 6: Keycloak role creation

Lastly, in Figure 7, Figure 8, and Figure 9 the user is created and configured with a password and role assignments.

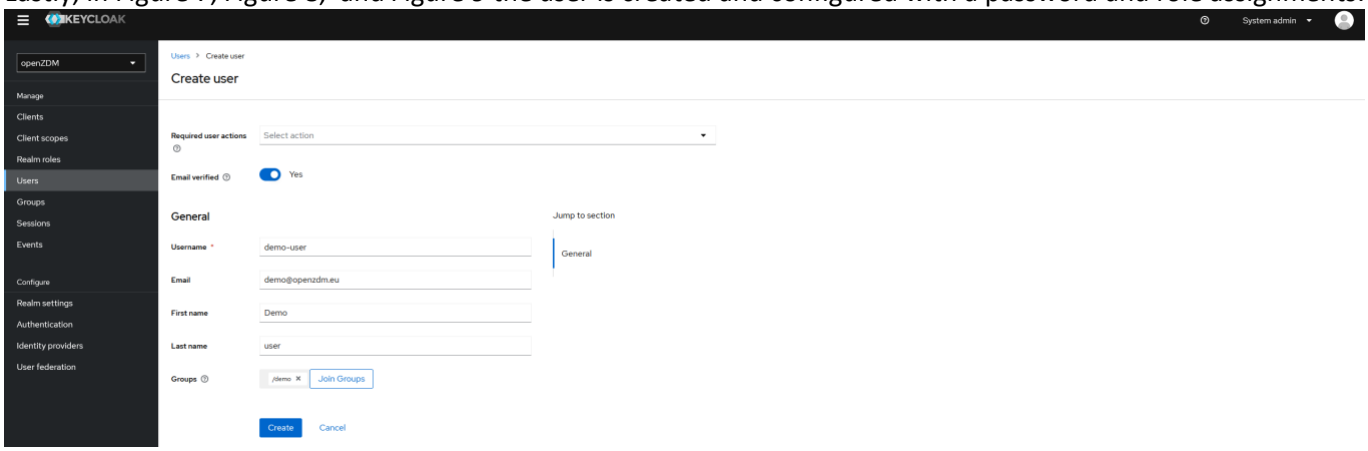


Figure 7: Keycloak user creation

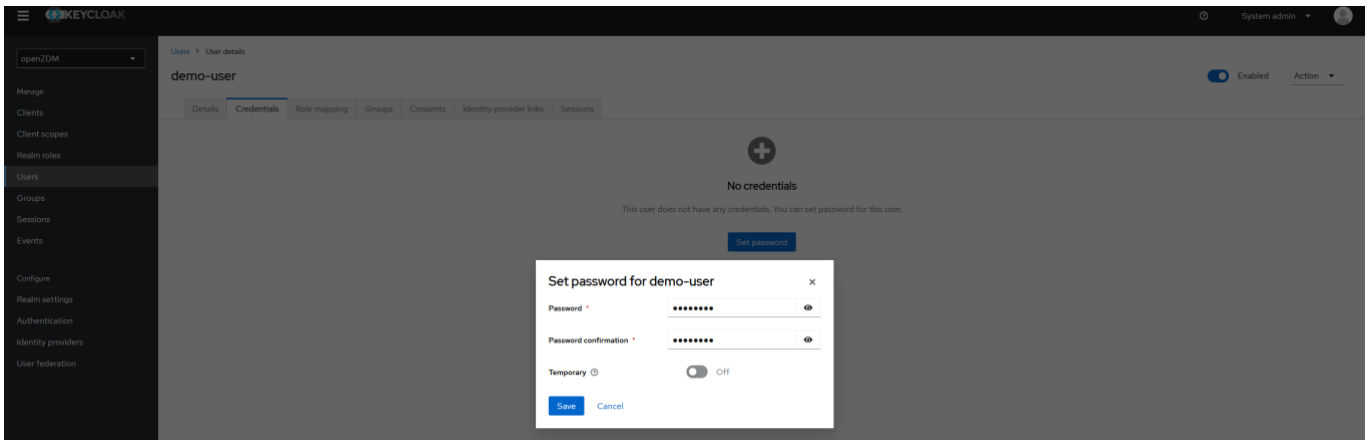


Figure 8: Keycloak password creation

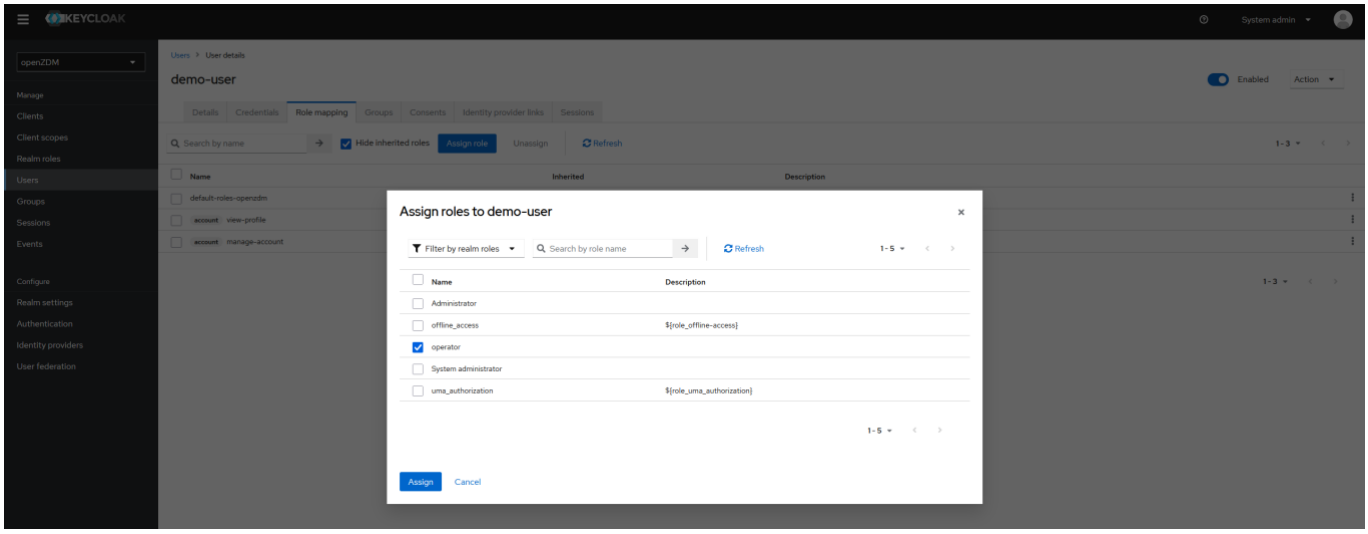


Figure 9: Keycloak role assignment

This process is used to configure all required groups, roles, and users for the proper operation of the platform. Moreover, this user can also create configurations for clients. Clients allow services to use the client and secret to retrieve tokens and use them to access other services inside the openZDM platform.

3.2 Applications

The next step is to configure the applications that this user will access when logging in. This is done from the User application preferences view (see Figure 10).

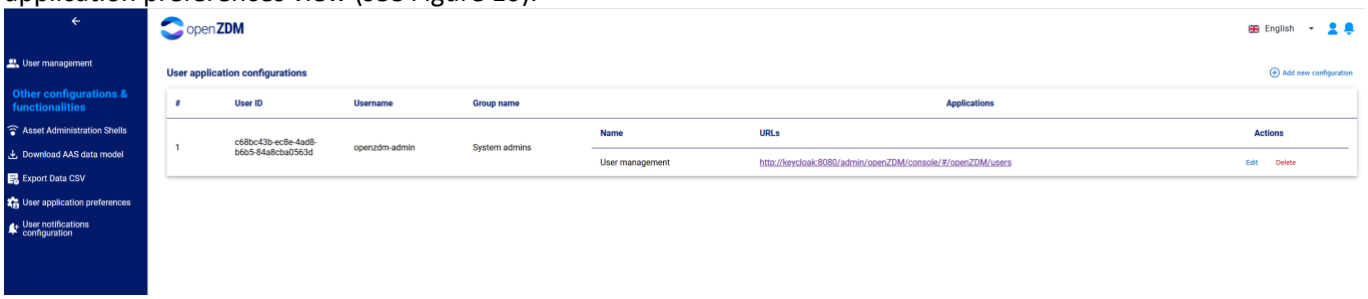


Figure 10: User application preferences view

The “Add new configuration” button at the top right corner of the screen leads to the appropriate view (see Figure 11). In this view, first the users are selected (in this case, only the demo-user is applicable). Next, whether the view will have one or two applications. In the case of one application, like in Figure 11, the user can select the type of application, like Digital Twin Toolset, provide the name, the URL, if the menu should be presented, and if this user should be directed to this application when logging in (starting screen). In the case that the menu is not presented, the application takes up the whole screen and provides its own menu buttons.

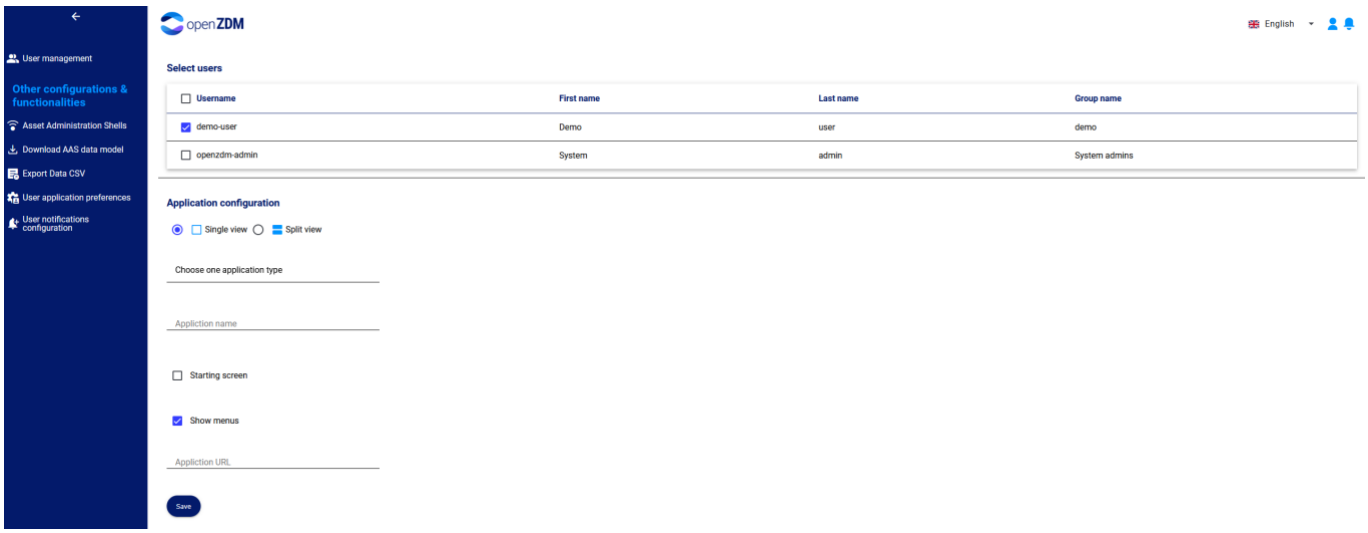


Figure 11: Application configuration for one or more user

When split view is selected then two views are required to be filled in from the user. The two views are arranged one on top of the other. The top view is presented in a smaller window and the bottom view is presented in a larger window. See a sample in Figure 12 where robot arm visualization is in the bigger window and the graphs in the smaller window.

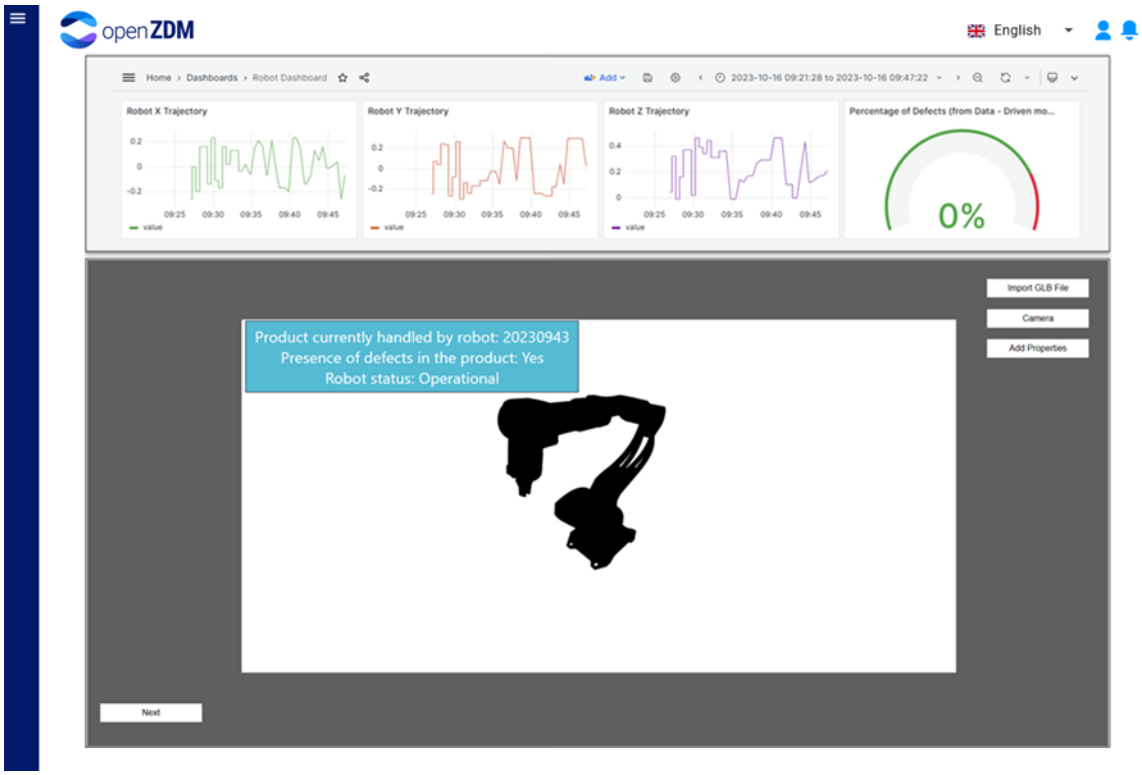


Figure 12: Split view arrangement

The form (see Figure 13) is the same as previously, with the addition of two more URLs. The first URL (bottom application URL) is to be presented in the large window. The second and third URLs are for the top application configured to be shown in a smaller window and the other in a big window, this was done to address scaling issues for applications that cannot auto-scale when the window size changes. Only one of these two URLs is mandatory.

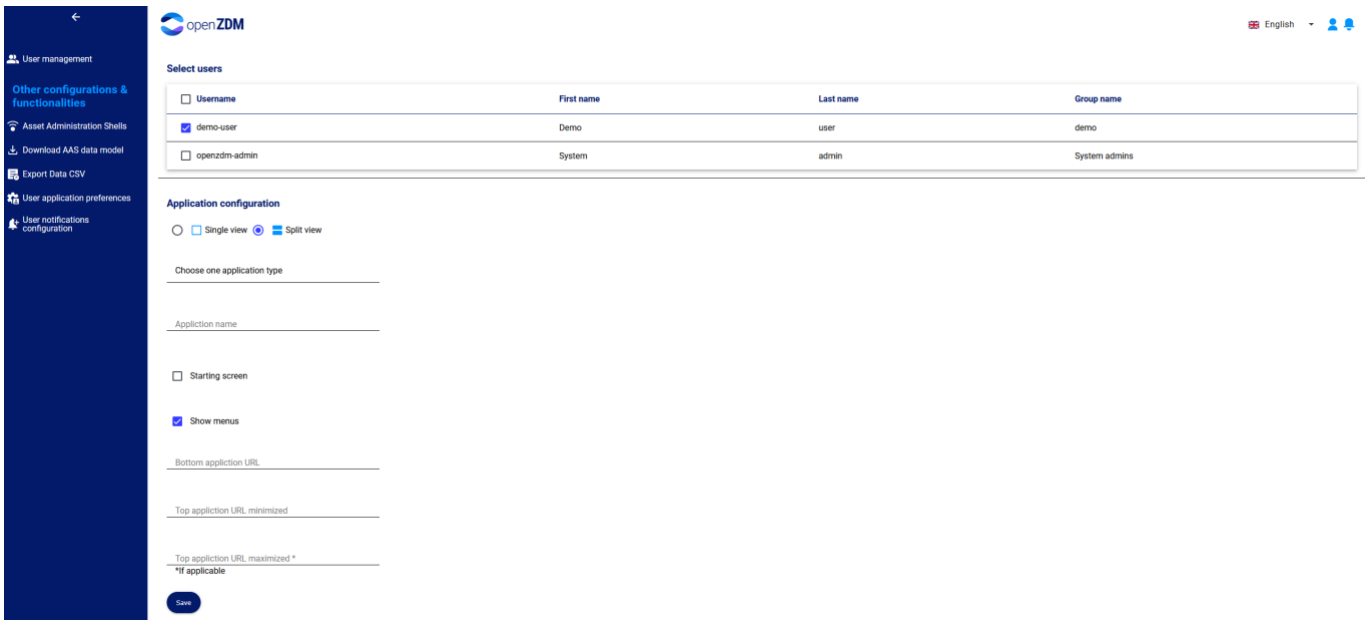


Figure 13: Application configuration for one or more user with split view

3.3 Notifications

The notification service of the openZDM platform is capable of sending notifications to its users through the user interface, SMS, and email. Notifications can be generated from other components of the platform using the notification service API or MQTT. The notification configuration workflow starts with pressing the button “User notifications configuration” and then clicking on the “Add new configuration” button, which leads to the form in Figure 14.

Figure 14: Notification configuration form

The user is selected. The ID is filled in automatically from the user’s profile in Keycloak. Then the type (User Interface or SMS, or Email) is selected. Then the configuration type is filled in. User-defined types are also possible to enable fine-grained notifications, such as when an NDI is not working. Lastly, the email and phone number are filled in.

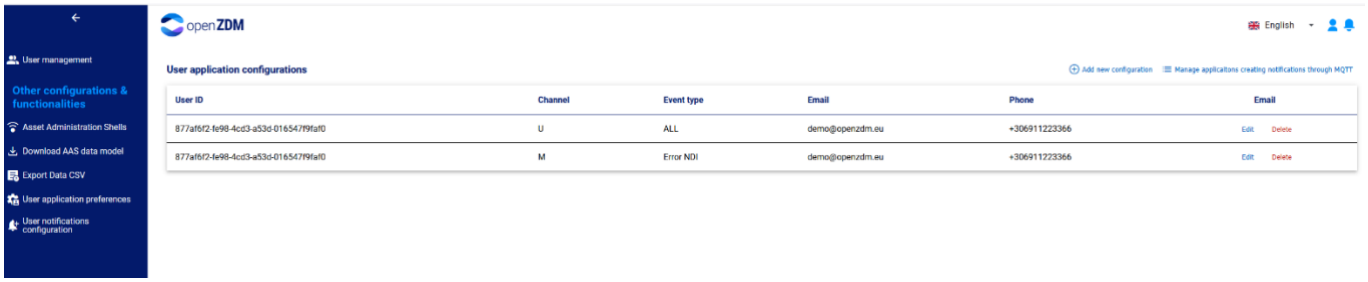


Figure 15: Configured notifications for demo-user

As depicted in Figure 15 multiple notifications are possible depending on the type and channel, as shown, demo-user will receive all notification types through the user interface and Error NDI through email. For services that use MQTT, an additional manual configuration is needed. The button “Manage applications creating notifications through MQTT” opens the dialogue for mapping service IDs and pilots (see Figure 16).

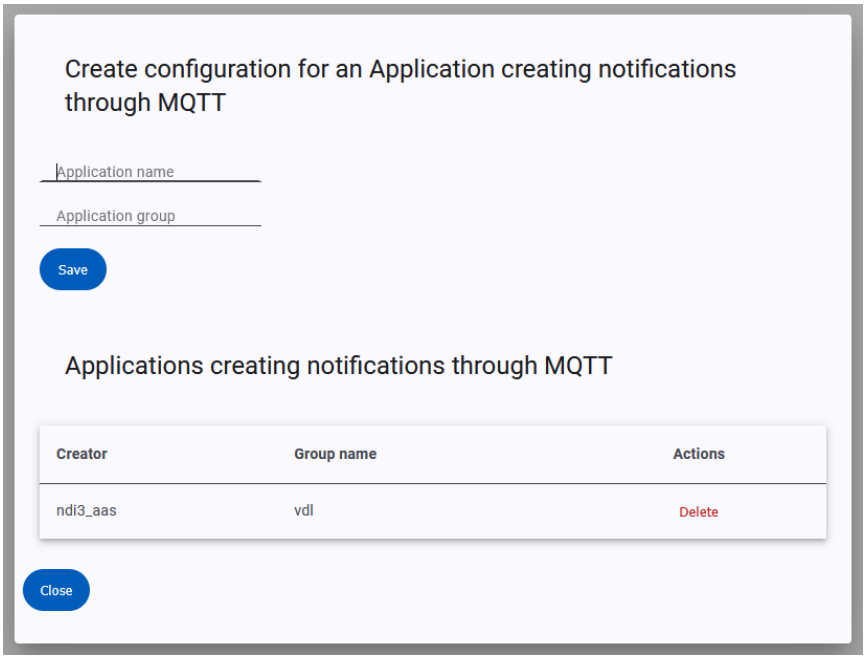


Figure 16: Notification MQTT mapping

3.4 Data export

The platform allows for historical data export for further analysis. The button “Export Data CSV” leads to the appropriate view. Then the button “Add new property” is used to select the AAS using its ID, the relevant Submodel, and the Submodel element (see Figure 17).

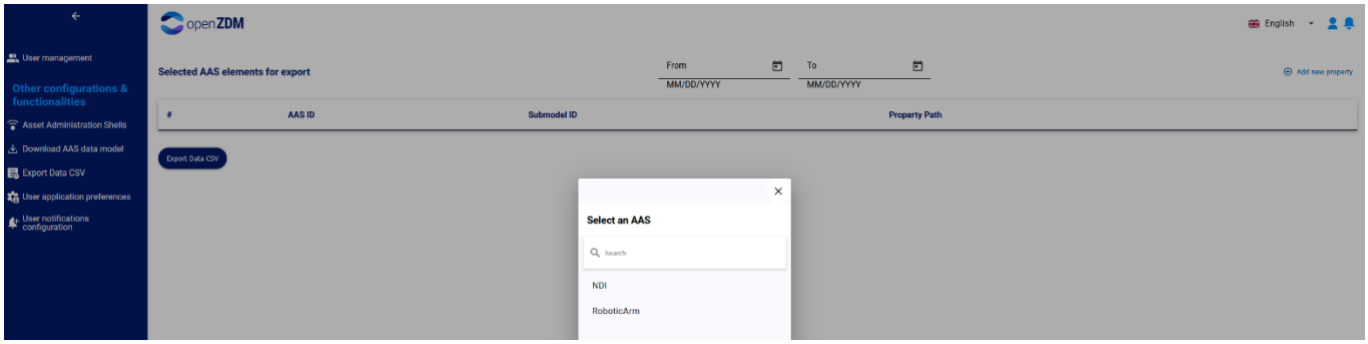


Figure 17: AAS selection for historical data export

As soon as the selection is complete the button “Export Data CSV” at the bottom of this view is used to export the file in CSV format (see Figure 18).

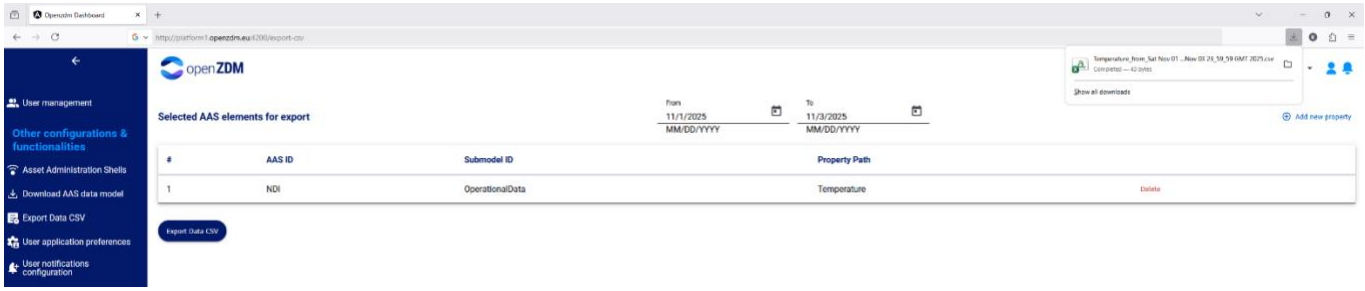


Figure 18: Excel downloaded

3.5 Digital Twin Toolset

To configure (create) a new digital twin, a user needs to undertake the steps presented in section 1.3. To present the steps for creating a new digital twin, a scenario where the digital twin of a robotic arm will be configured. To configure it, the following prerequisites should be established:

1. The user who configures the digital twin should have the digital model in 2D or 3D of the robotic arm
2. The AAS of the robotic arm should be available to the openZDM platform
3. Synthetic or real-world data of the robot is being fed to the openZDM platform
4. Synthetic or real-world historical data of the robot being available in the case of a data-driven modelling approach is selected.

Initially, the user of the tool configures the name of the digital twin and selects a use case already registered to the opennZDM platform that the digital twin belongs to. This initialisation is followed by the upload of the digital model of the robotic arm. In the specific scenario, a 3D model is provided to the toolset using a GLB file. This process is illustrated in Figure 19.

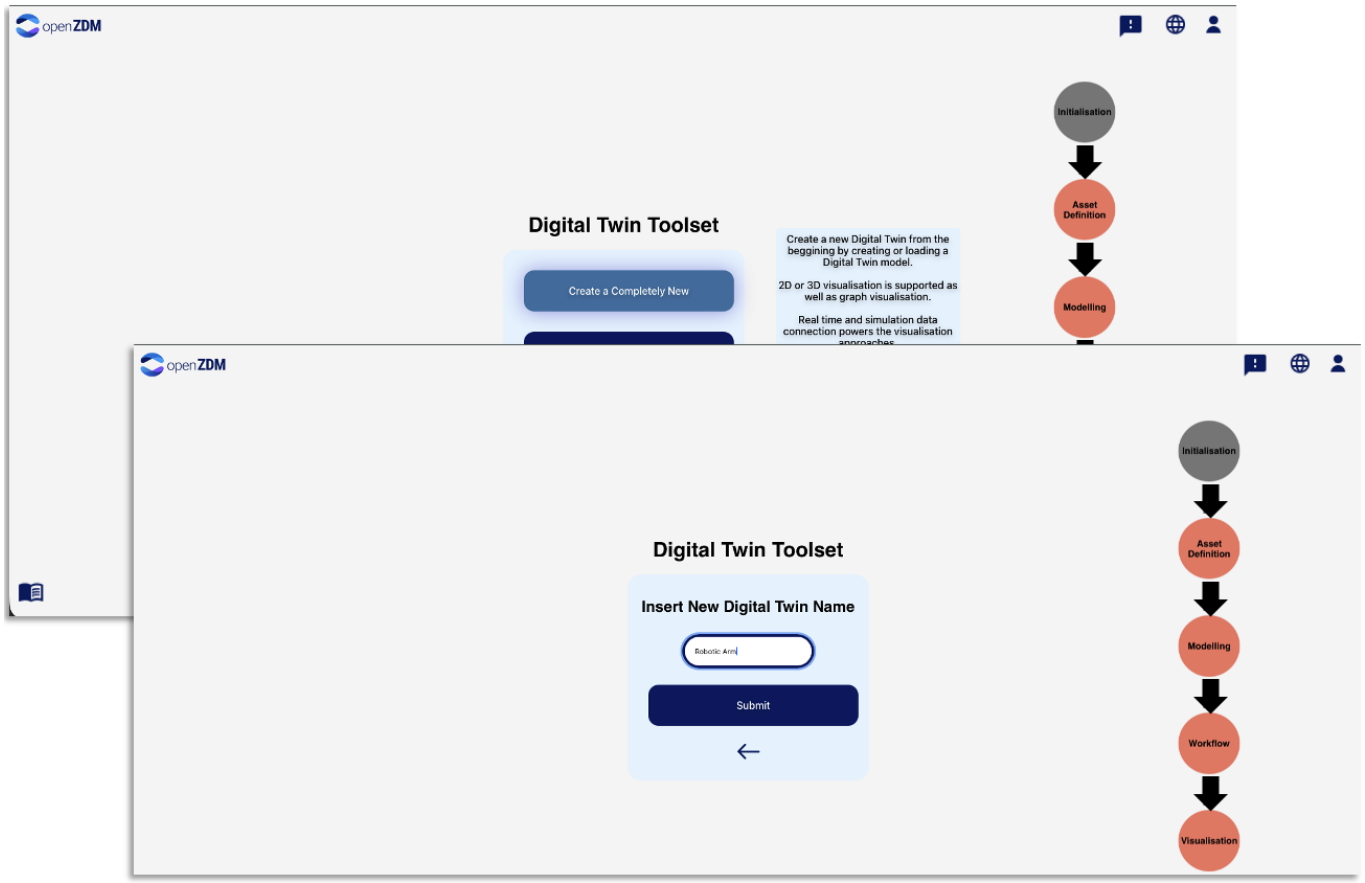


Figure 19: Initialisation of the DTT

Following the initialisation and asset definition, the user enters the modelling stage. To simplify the process, a data-driven model will be created. The data-driven model will simulate the movement of the robot that performs a

screwing operation in relation to the generation of defective parts. To achieve this, the user is exposed to possible data-driven models such as RFR (Random forest regression), SVR or LSTM (Long Short-Term Memory recurrent neural network). Out of the three, the RFR option is selected due to its ease of use and due to the lower computational requirements needed for inference. This process is illustrated in Figure 20.

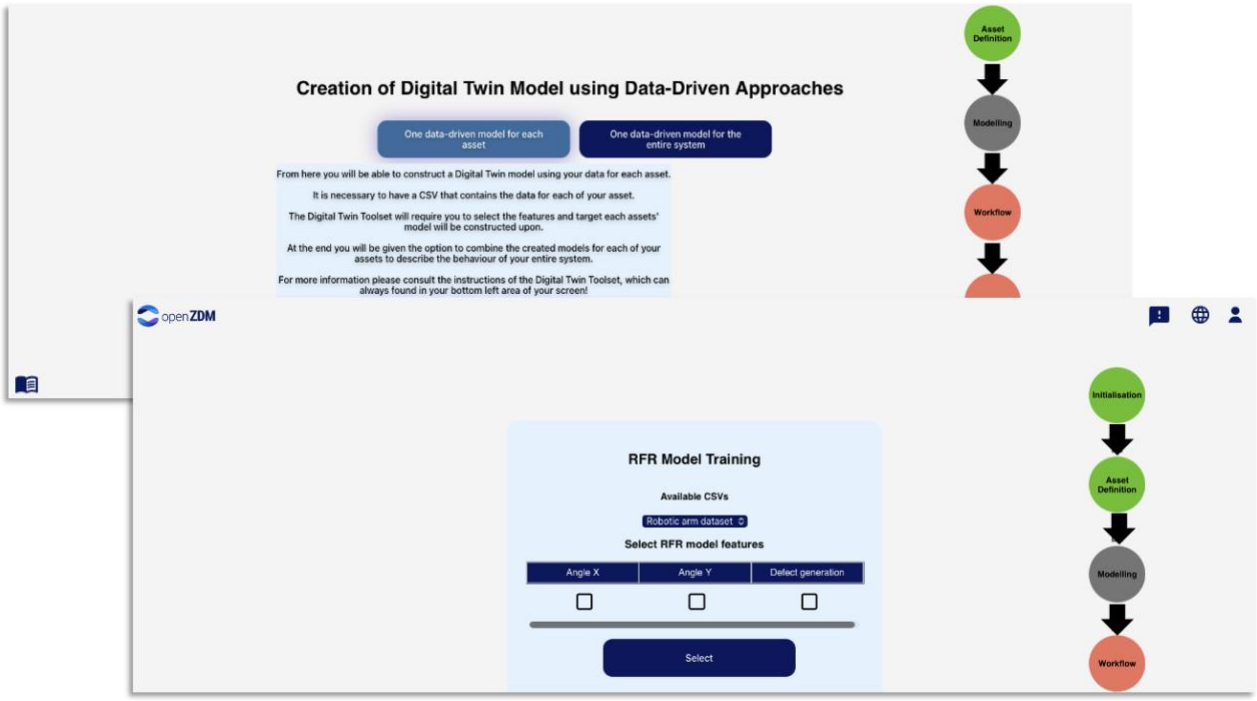


Figure 20: Configuration of the digital twin model in the DTT

Upon the modelling is completed successfully, the user is exposed to the workflow creation engine. Here the user via a Node-RED instance is able to provide inputs to the simulation model created in the previous step as well as connect its outputs with the databases exposed by the toolset or any available analytics that run on top of the digital twin. This process is illustrated in Figure 21.

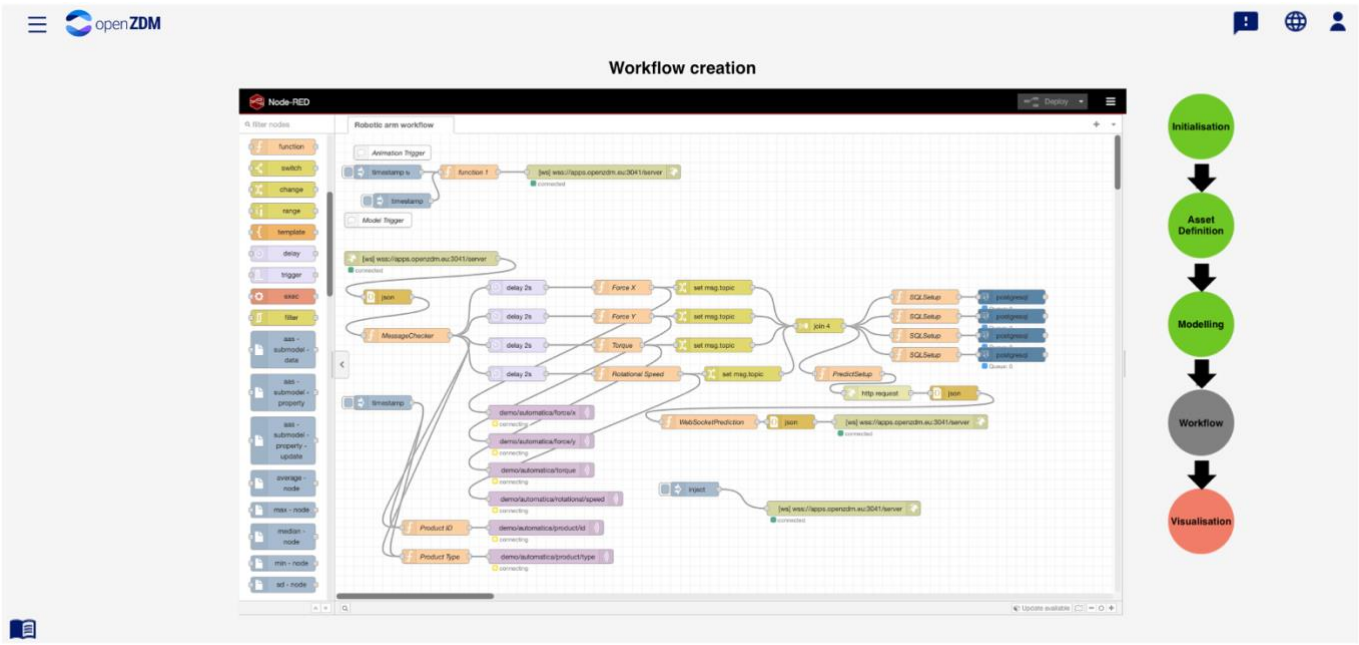


Figure 21: Configuration of the digital twin workflow in the DTT

Lastly, the user is directed to the visualisation service. Since a 3D visualisation was selected during initialisation, a 3D canvas is exposed to the user that allows them to place the robotic arm on the scene and connect it with MQTT

triggers for animations and information panels for real-time data visualisation on top of the scene. This process is illustrated in Figure 22.

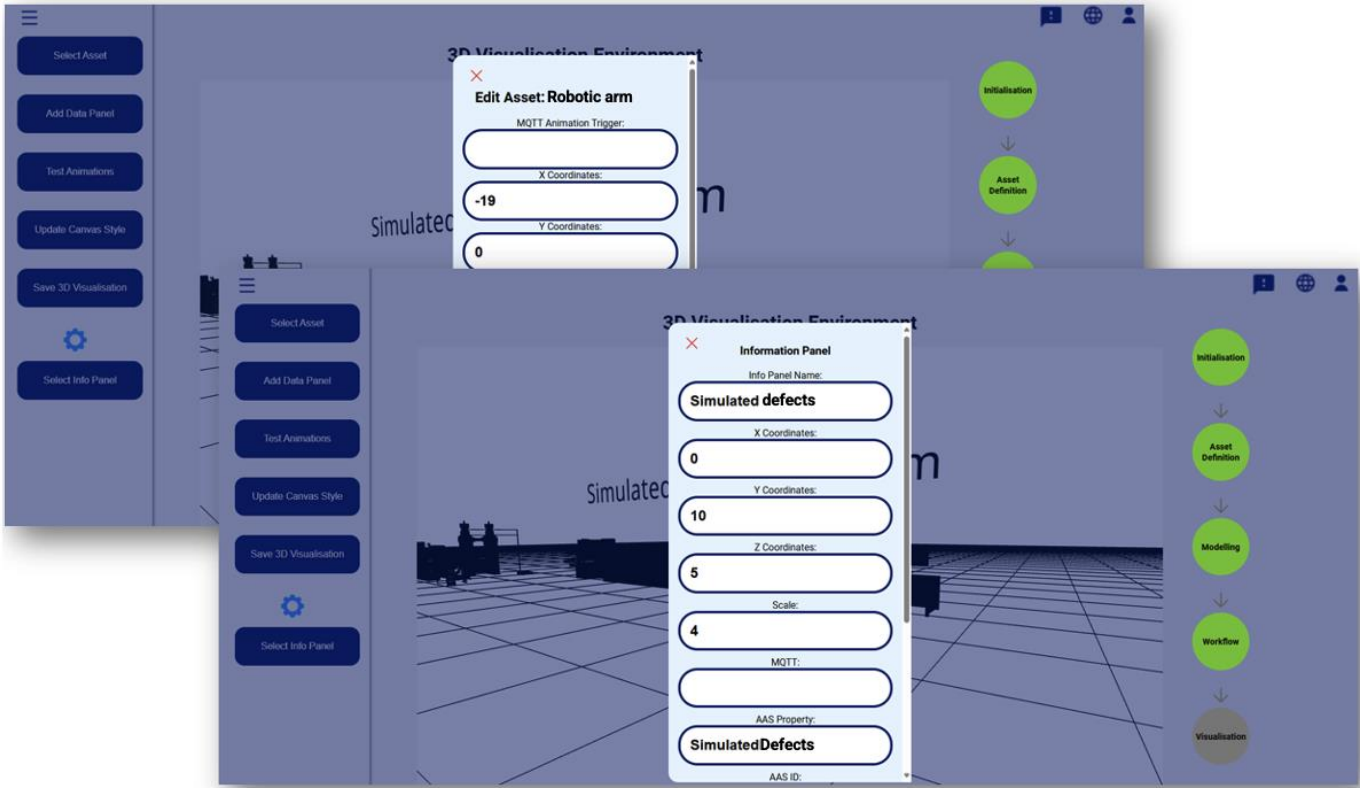


Figure 22: 3D visualisation configuration in the DTT

3.6 Configurator

The Configurator provides a visual, low-code environment to build and deploy containerised architectures. It responds to the unstoppable trend in the market where new applications are built from the start with microservices and where legacy systems are migrated to new architectures or are integrated into them. Its interface is based on a canvas-style editor, allowing users to deploy complete microservice (Docker) based architectures by visually combining components and defining their interactions. The tool automates the generation of deployment files and manages dependencies, networking, and runtime configurations, simplifying complex orchestration tasks.

Users access the Configurator through a web-based canvas to create a new project or edit an existing one. Using the visual editor (see Figure 23), users drag and drop components, representing services, data sources, or processing units, onto the canvas and define their interconnections (MQTT, HTTPS, etc.). Configuration panels allow users to adjust component parameters, environment options, deployed on one or multiple hardware devices. The tool validates the configuration to maintain consistency and correctness across the entire application.

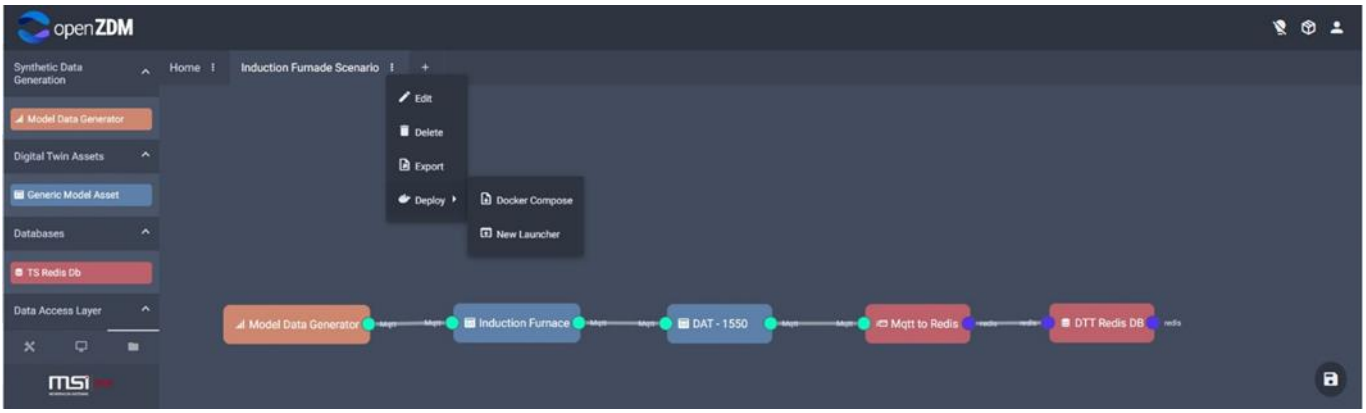


Figure 23: Creation and deployment of applications using the Configurator

Once configuration is complete, the Configurator automatically generates the required deployment artefacts, mainly a *docker-compose.yml* file and related configuration files. Deployment can be executed in two ways:

- Manual deployment, by exporting the generated artefacts for external use.
- Automated deployment, using a Launcher Agent that retrieves the application definition from the backend and manages the startup or update of services on the target infrastructure.

After deployment, users can monitor their applications in real time via the integrated dashboard (Figure 24), which provides access to logs, performance metrics (via Prometheus), and notifications related to the running services.

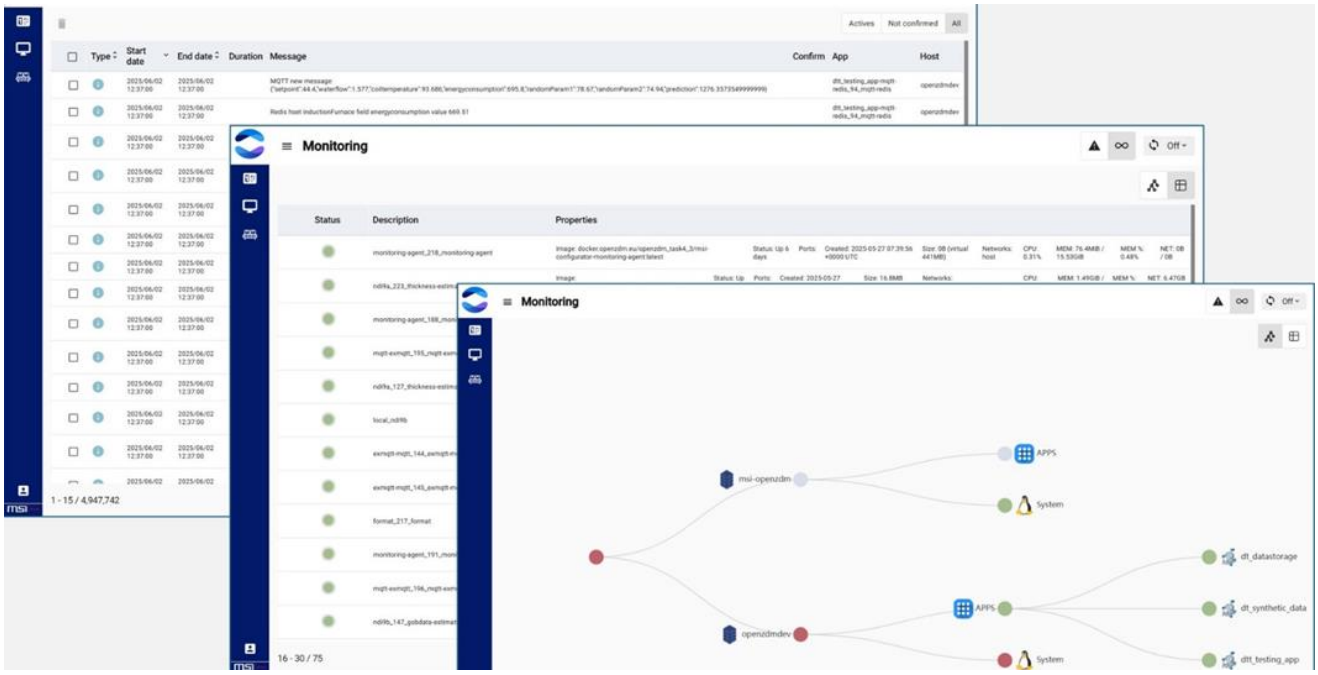


Figure 24: Monitoring of applications using the Configurator

Through this process, the Configurator transforms application deployment into a simple and guided visual workflow. Users can configure, deploy, and monitor multi-service containerised applications with ease, ensuring repeatable and reliable deployments across all openZDM environments.

3.7 Quality Assessment Modules

The configuration of modules focuses on training models via the training app and deployment via Configurator.

Figure 25: Configuration of the name and methodology of a new DAT in the DAT Training App

3.7.1 DAT Model Training Application

The DAT Model Training Application, hereafter referred to as the DAT Model Training App, allows the training of new DAT modules. Additionally, the DAT Model Training App allows the update of existing DAT modules. The configuration of a new DAT begins with entering the unique name of the DAT module that should be created. Also, the methodology of the DAT is selected at this stage. The App supports the four methodologies listed in deliverable "D4.2 - Methodologies and first implementation", including pre-processing, monitoring, prediction & classification, as well as descriptive & prescriptive. This process of initialising a new DAT is illustrated in Figure 25.

Following the initialisation, the user uploads their data to the DAT Model Training App and selects a model that fits in the methodological category selected during the first stage of the DAT creation. Supported data formats include CSV, Excel, Point Clouds and ZIP files in the case of an image-based DAT being created. This process is illustrated in Figure 26. This step is then followed by the configuration of the model hyperparameters. Each model exposes its unique

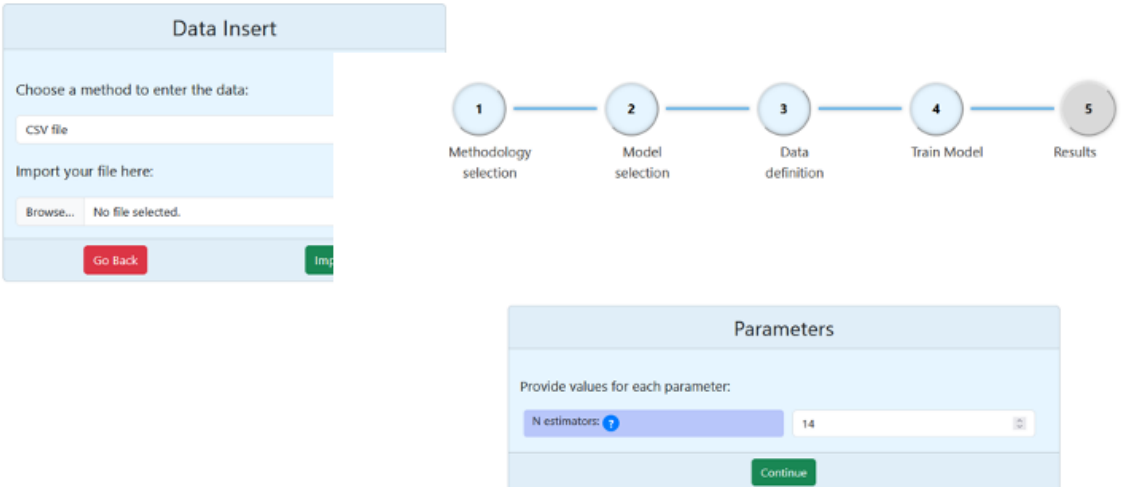


Figure 26: Configuration of training data and hyperparameters in the DAT Training App

hyperparameters that are available for tuning. Upon definition of the hyperparameters, the user trains the model through the application, and its error metrics are presented following the model validation stage. In case the user deems the performance acceptable, the model can be published in an automated way to the openZDM Docker Harbour, and it is registered to the openZDM Eureka service. This is also illustrated in Figure 26.

3.7.2 Deployment of Data Analytics Tools

As outlined in Section 1.4, the DATs are key components of the openZDM platform, providing AI-driven data analytics. This section describes how the DATs are deployed, configured, integrated, and made operational within the openZDM environment.

As previously discussed, the configurator enables this deployment process by offering a visual, low-code interface that allows users to build and deploy containerised architectures with minimal effort.

As an example, Figure 27 shows the configuration used to deploy a DAT (e.g., DAT#11 for rule-based monitoring production processes) through the configurator. The components (rectangular nodes in the figure) are provided as NPM packages and can be customised as needed. Users can define data queues, databases, ports, and inter-component connections, automatically generating the required dependencies.

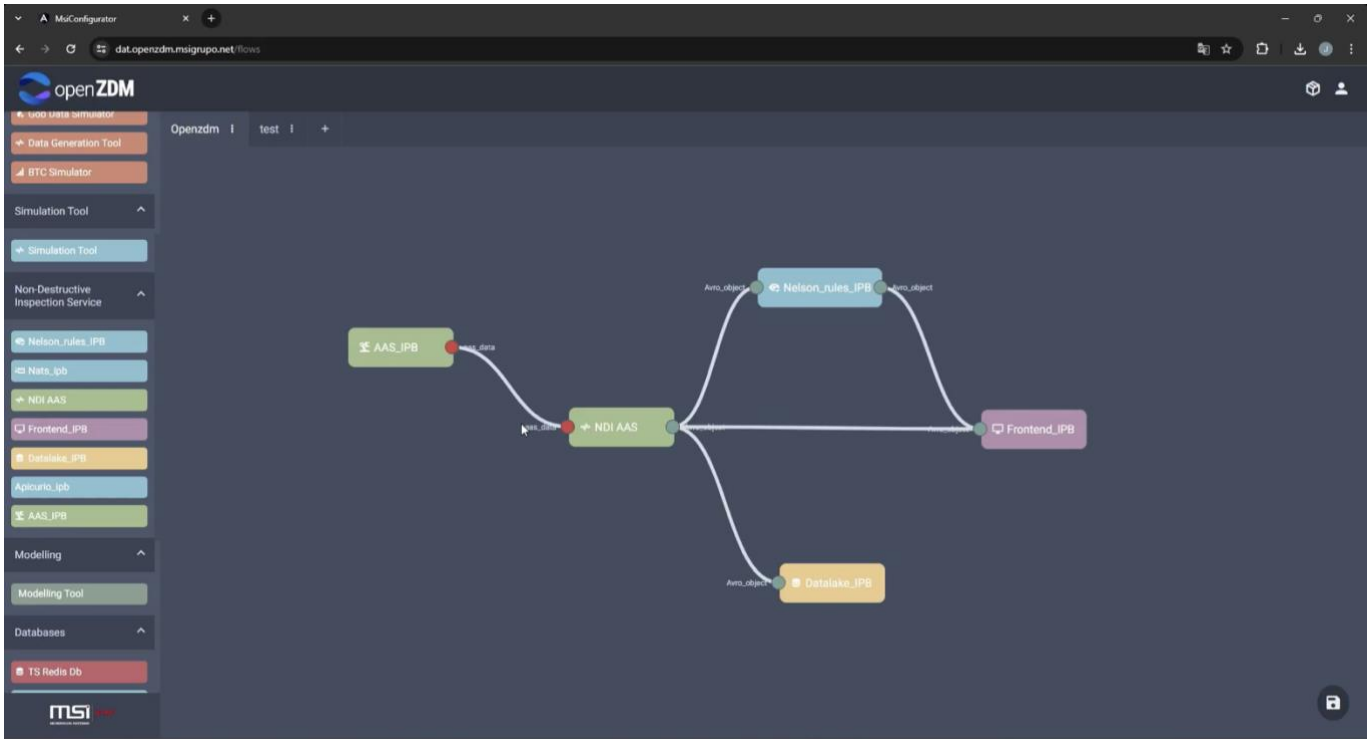


Figure 27: Configuration example used to deploy a DAT

In this example, the flow schema used to deploy DAT#11 includes:

- AAS component: enables to collection of data from the shop floor.
- NDI AAS component: formats and transmits data in AAS format.
- Datalake component: stores production data.
- Nelson_rules component: analyses the data, generates alerts, and sends them to the frontend.
- Frontend component: receives data in real time and displays it to the user.

Once configured, the system exports a ready-to-use Docker Compose file, significantly reducing deployment time. After downloading it, deployment only requires running a single command (e.g., “docker compose up –d"), as the remaining configuration is already handled during the DAT setup. After this step, the DAT is fully operational.

The deployment process for other DATs follows the same approach, differing only in the specific components used according to the functionality of each DAT. Therefore, it will not be covered here.

3.8 Decision Support Tool

The DST relies on the presence of a manufacturing system simulation model that is created in the DTT (for additional information, refer to the modelling step of the DTT in section 3.5). The DST relies on asset information and controllable parameters from the DTT that are defined in the workflow engine. Thus, the sole customisation the DST requires is the definition of the DTT URL that the DST utilises to query the DTT APIs. Such a configuration is carried out through the DST’s docker-compose.yml file.

4 Platform operation

4.1 Landing page

By default, all users are forwarded to the landing page (see Figure 28). There is an option to override this behaviour as detailed in section 3.2. The landing page shows the Processes and NDIs monitored, and the applications configured.

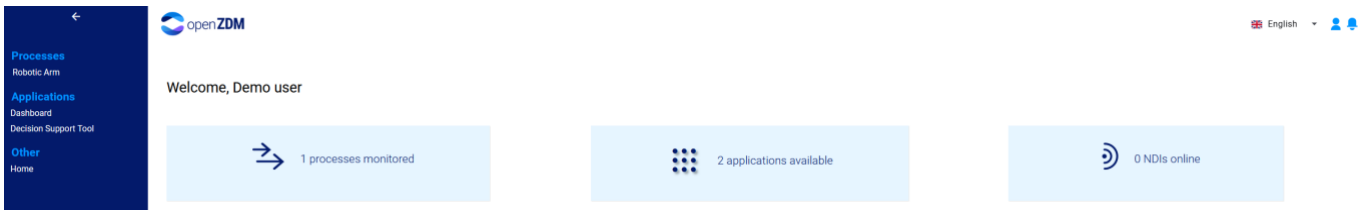


Figure 28: openZDM User Interface landing page

4.2 Process and NDI views

If processes or NDIs are registered as AAS for the pilot (group) of the user, they will appear in the left side menu under the appropriate category. When clicking on one of these elements, the default behaviour of the user interface is to illustrate the latest values from the OperationalData submodel of the AAS (see Figure 29).

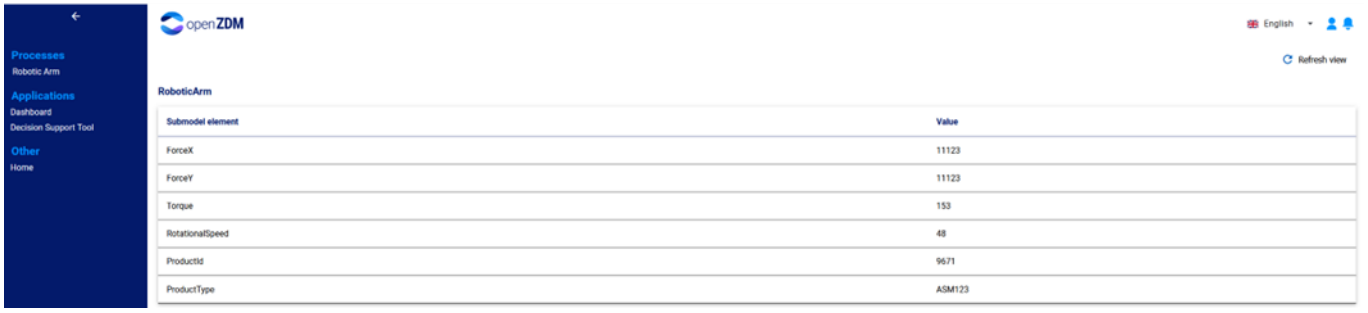


Figure 29: OperationalData submodel values

4.3 Notifications

As notifications come in from the openZDM applications, they are available via the user interface (see Figure 30). A dedicated drop-down menu is provided for viewing the latest unread notifications.

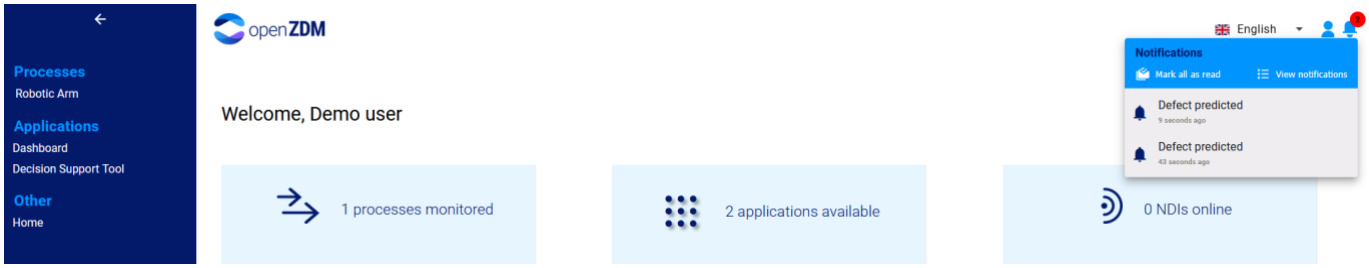


Figure 30: openZDM platform notifications in the landing page

In addition, a separate view is provided to view all past notifications (see Figure 31).

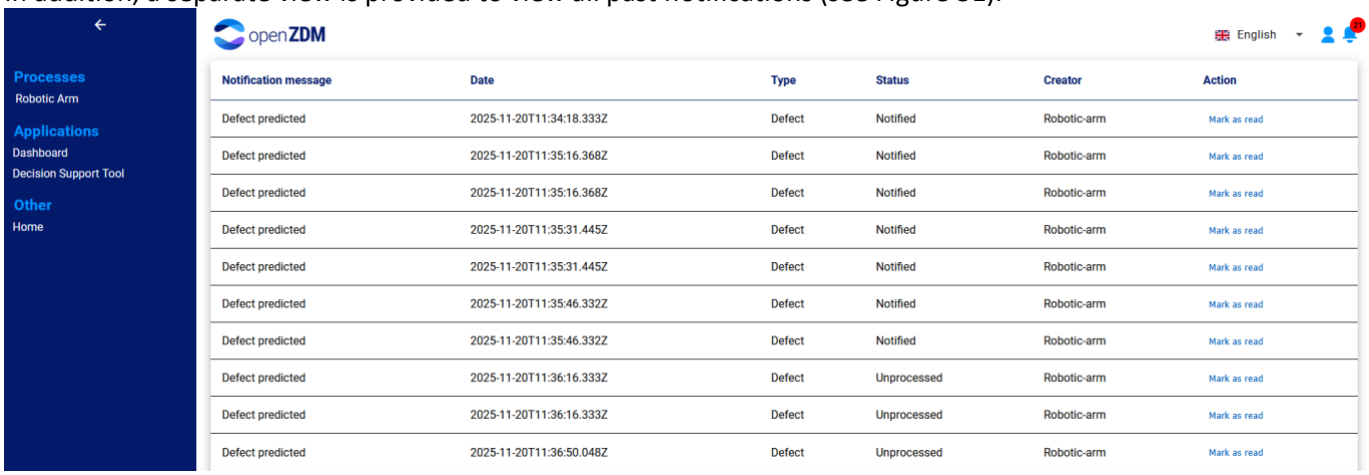


Figure 31: Notifications view

4.4 Application views

As detailed in section 2, all user interfaces of the openZDM applications are integrated with the main user interface of the platform and are available through it, as illustrated in Figure 32. In the remainder of this section, the screenshots of the applications may be cropped to illustrate specific parts of a user view to enhance clarity.

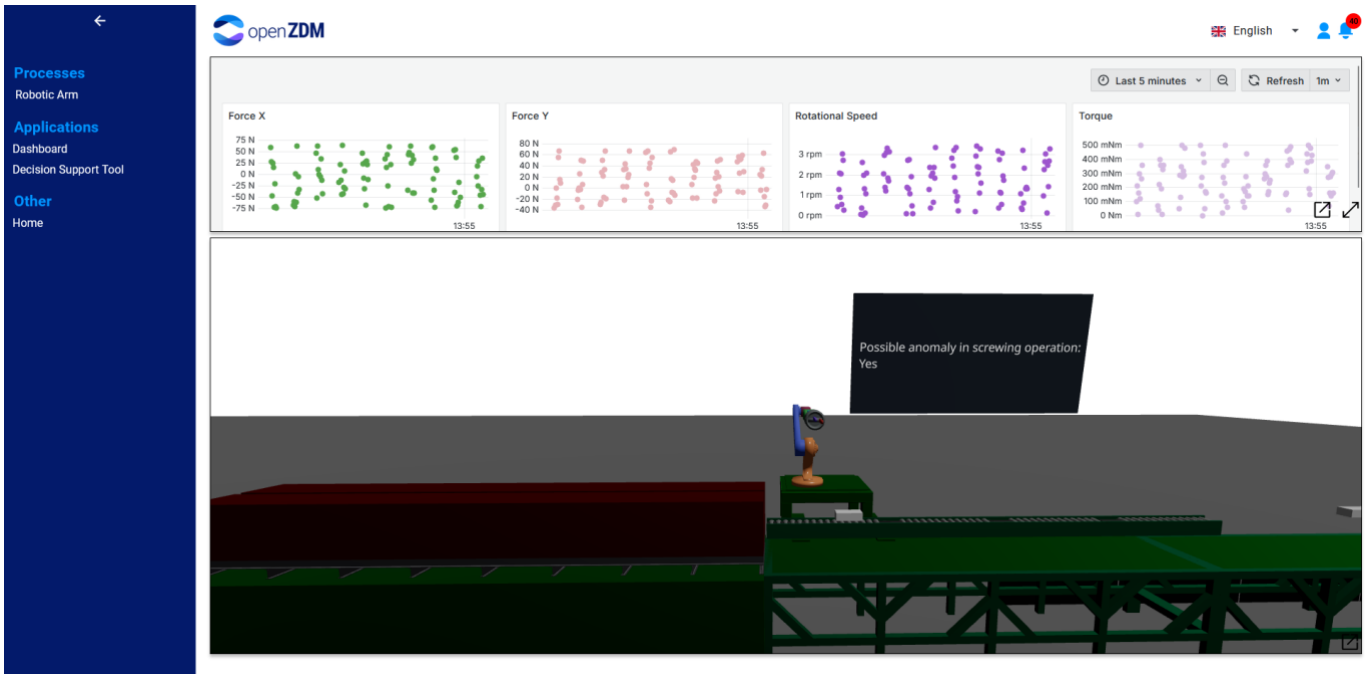


Figure 32: Viewing an openZDM application through the main user interface

4.4.1 Digital Twin Toolset

For the platform operation, the same demo used during the configuration process presented in section 2 is being used to demonstrate the Digital Twin Toolset operation. During operation, the DTT exposes the visualisation canvas saved during the configuration stage. In the specific case, the 3D model of the robotic arm is animated, with animations being triggered based on the MQTT animation topic defined. In addition, real-time information for defects being predicted is shown in the information panel rendered on top of the canvas. This is illustrated in Figure 33 and is demonstrated in the video of Section 4.5.

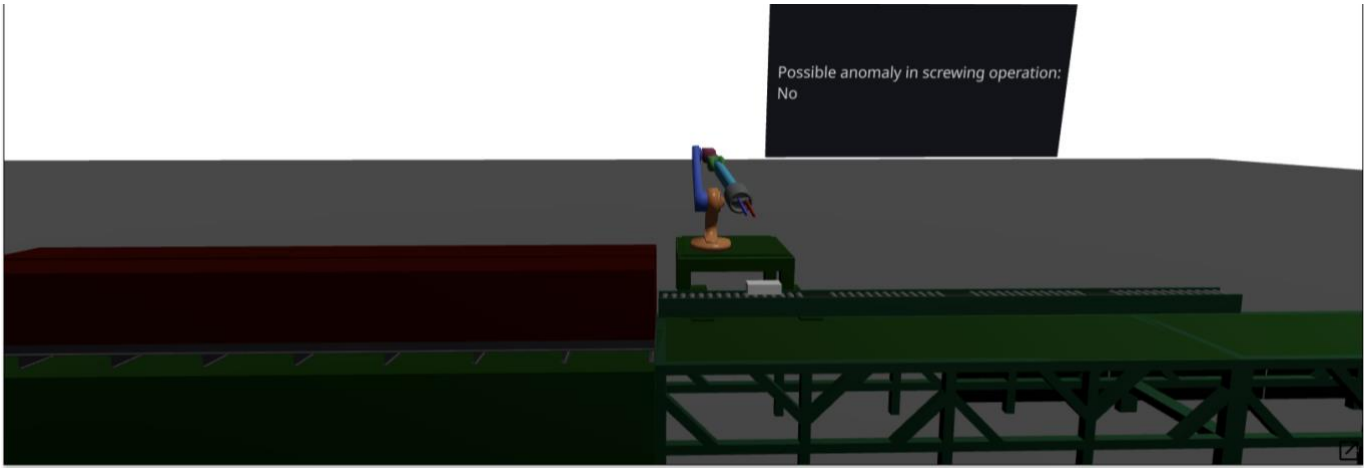


Figure 33: Multi-target quality assessment module for dimensional predictions

4.4.2 Quality Assessment Modules

As previously discussed in Section 1.4, DATs play a critical role within the platform, particularly in performing AI-based data analytics to enable monitoring and the early and real-time detection of process and product quality

deviations and trends. In this context, this section aims to present a brief view of the operation of the implemented tools, highlighting their main functionalities.

4.4.2.1 Quality assessment module for predicting product dimensional properties – DAT#0

This quality assessment module is capable of performing multi-target predictions of core product dimensions in large metal bars. Specifically, it can predict the width, thickness and straightness over width of metal bars. In addition, the DAT#0 exposes a user interface for real-time visualisation of inference results that operators and engineers are exposed to. In addition, the user interface allows engineers to navigate to older products using their unique product identifier, such as their product ID. Such functionality is illustrated in Figure 34 and demonstrated in the video found in section 4.4. Lastly, on top of DAT#0 an explainable AI layer (extension of an implementation for image-based explainability as reported in the 2nd periodic report of openZDM) can be connected in order to enable better understanding on the root-cause of defects for human operators. In particular, the layer highlights the importance of different production parameters in the predicted defect related to the monitored and anlyzed dimensional properties. This in term enables real-time quality inspection and thus, control.

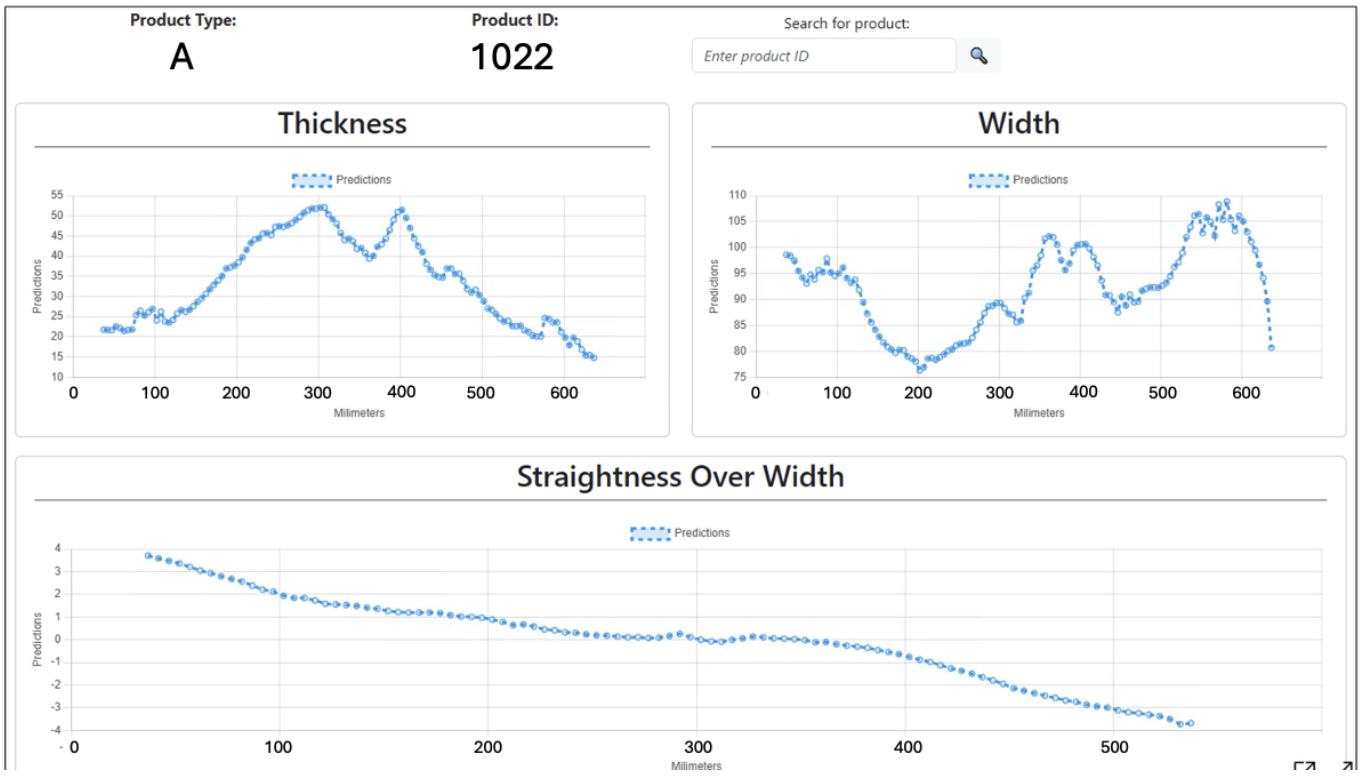
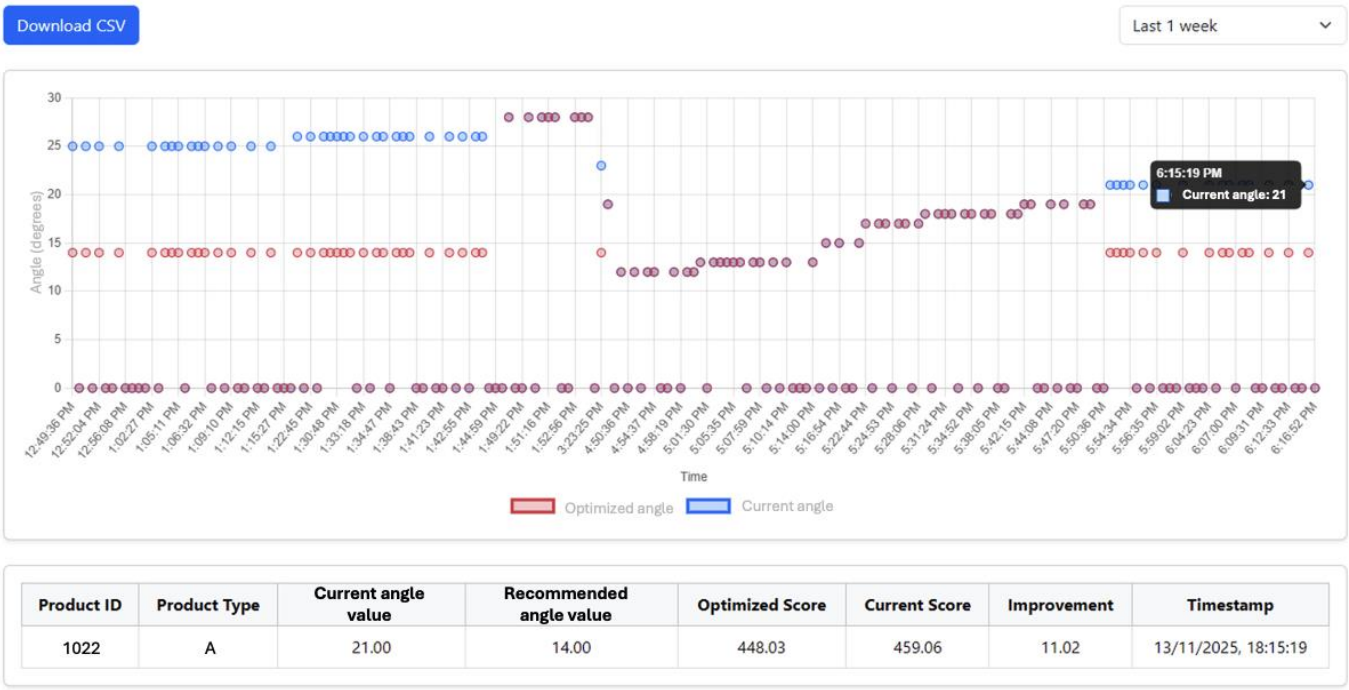


Figure 34: Multi-target quality assessment module for dimensional predictions

4.4.2.2 Quality assessment module for identifying anomalies in steel metal processing – DAT#1

This quality assessment module is based on anomaly detection and optimisation algorithms to advise operators in altering specific production control parameters, targeting the prevention of the next product being defective. During inference, operators in the line are exposed to a user interface that allows them to visualise the current setting of the control parameter together with the suggested correction. Enabling trustworthiness and auditability of the quality of the recommendations, managers can visualise past recommendations, enabling feedback provision to developers of the quality assessment module. The user interface functionality is illustrated in Figure 35 and demonstrated in the video found in section 4.4.



Product ID	Product Type	Current angle value	Recommended angle value	Optimized Score	Current Score	Improvement	Timestamp
1022	A	21.00	14.00	448.03	459.06	11.02	13/11/2025, 18:15:19

Figure 35: Recommendations produced by a quality assessment module based on anomaly detection and optimisation

4.4.2.3 Quality assessment module for calculating the environmental impact of systems – DAT#2

This quality assessment module targets the monitoring of environmental aspects of a manufacturing system based on Life Cycle Inventory Data (LCI Data). Given the complexity of acquiring real-time data from various manufacturing assets, such as their energy consumption, their scrap generation, or raw material use, the specific module performs calculations based on user input to their system's LCI Data. The module performs a Life Cycle Assessment (LCA) of the manufacturing system in accordance with ISO 14040. It exposes to a user interface both the update functionality of LCI Data as well as the calculations for climate change (Global Warming Potential), ecotoxicity in freshwater and human toxicity. Such aspects are illustrated in Figure 36 and demonstrated in the video of section 4.4.

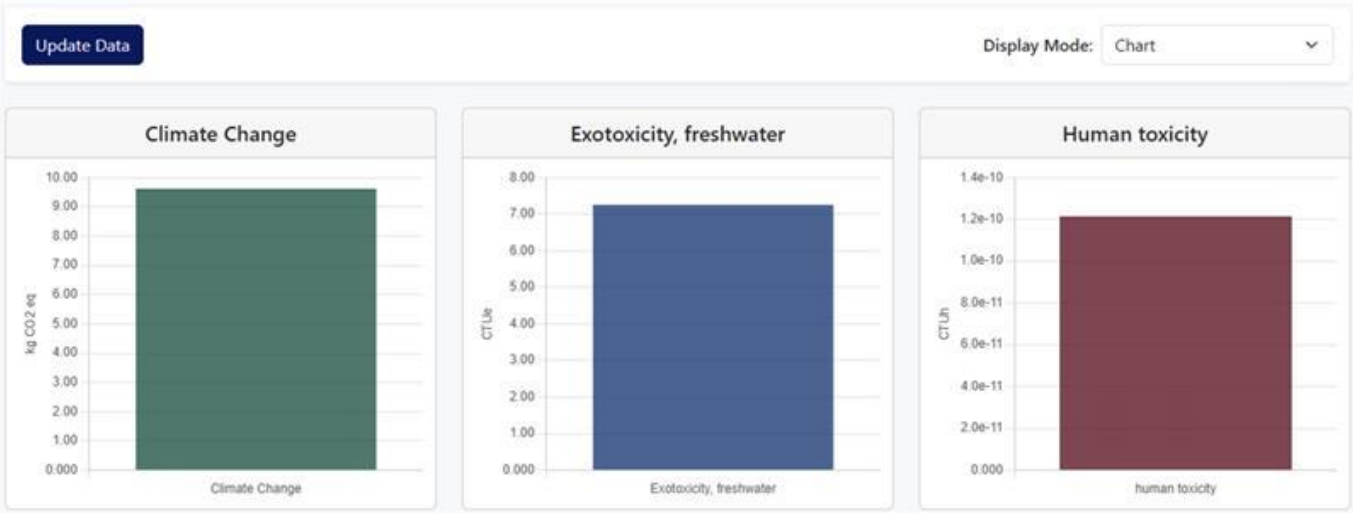


Figure 36: Environmental impact calculations via a quality assessment module utilising LCA

4.4.2.4 Quality assessment module for real-time defect prediction and explanation – DAT#3

DAT#3 is the core module for real-time defect management, which integrates prediction, explanation, and prevention.

- Defect Prediction & Explanation: The process begins when DAT#3 receives real-time data for a new panel. It uses a high-performance XGBoost model to perform a two-stage classification: first, it predicts if a defect will occur, and if so, it then identifies the specific defect type. Simultaneously, it uses SHAP to explain the "why"

behind the prediction, identifying the top process variables (e.g., temperature, pressure) responsible for the potential defect (see Figure 37).

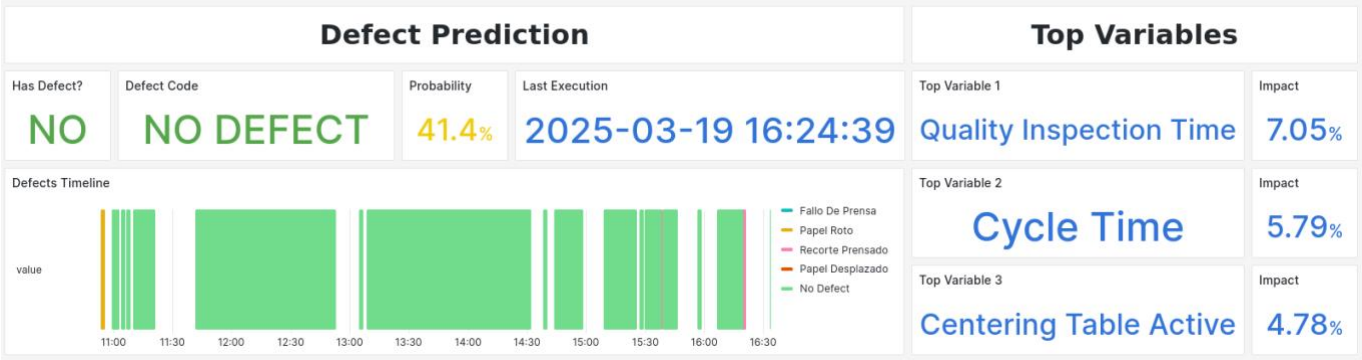


Figure 37: DAT#3 dashboard with defect prediction and explanation

- Defect Prevention & Recommendation: When a high probability of a defect is predicted by DAT#3, it automatically triggers the defect prevention functionality. This component uses optimisation algorithms to compute and recommend optimal machine parameter adjustments. This actionable recommendation is sent to the operator, allowing them to adjust the machine settings and proactively prevent the defect from occurring in subsequent panels (see Figure 38).

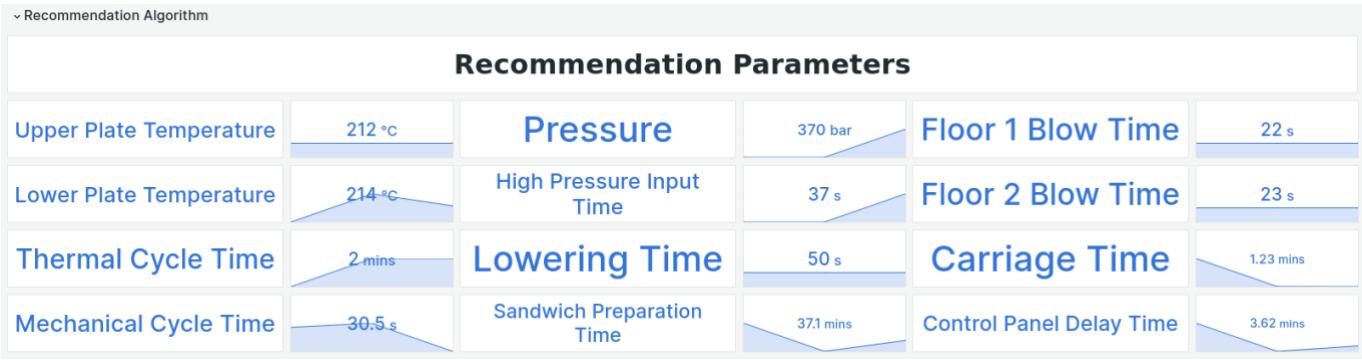


Figure 38: DAT#3 dashboard with machine parameters recommendation

4.4.2.5 Quality assessment module for new product simulation – DAT#4

DAT#4 is a specialised module designed to minimise waste during the setup phase for new panel types for which no historical production data exists. Its primary function is to generate optimal initial machine settings (a "recipe") by using a Zero-Shot Learning (ZSL) approach. The process is as follows:

- Estimate Defect Ratio: When a new product is introduced, the module first estimates its potential defect ratio based on its characteristics, even without prior data.
- Recommend Initial Parameters: Using this estimation, DAT#4 recommends an initial set of machine parameter configurations (e.g., pressure, temperature, speed).

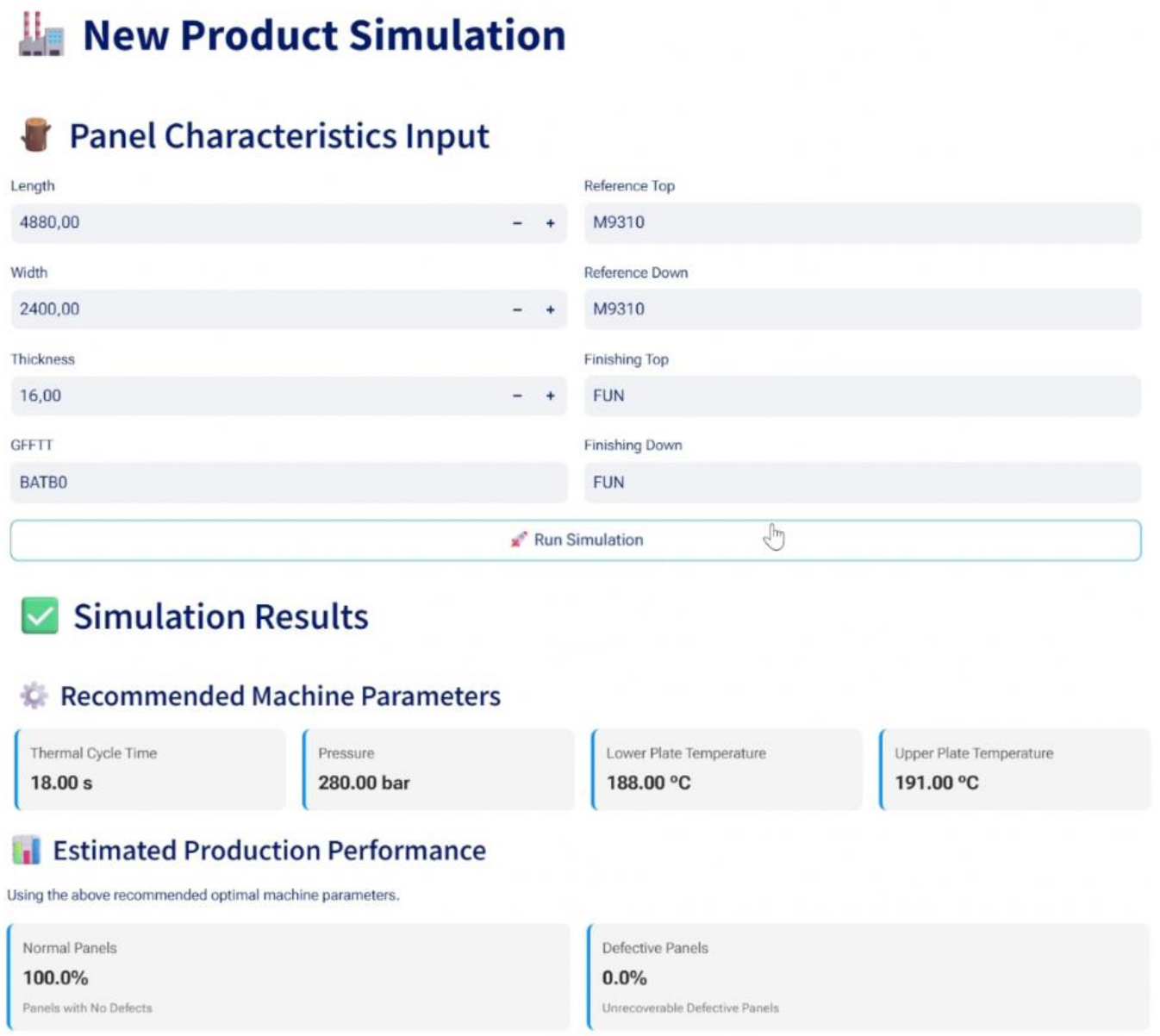


Figure 39: DAT#4 dashboard for the simulation of new products

This allows operators to start production of a new panel type with an optimised setup, significantly reducing the material and time (ramp-up) that would otherwise be wasted on trial-and-error configurations (see Figure 39).

In addition to single product analysis, the module also supports a batch simulation feature. An operator can provide an Excel file containing a list of multiple new production order codes. The system will then process this file, simulating each order to efficiently estimate their respective defect ratios and generate recommended settings for all of them at once (see Figure 40).

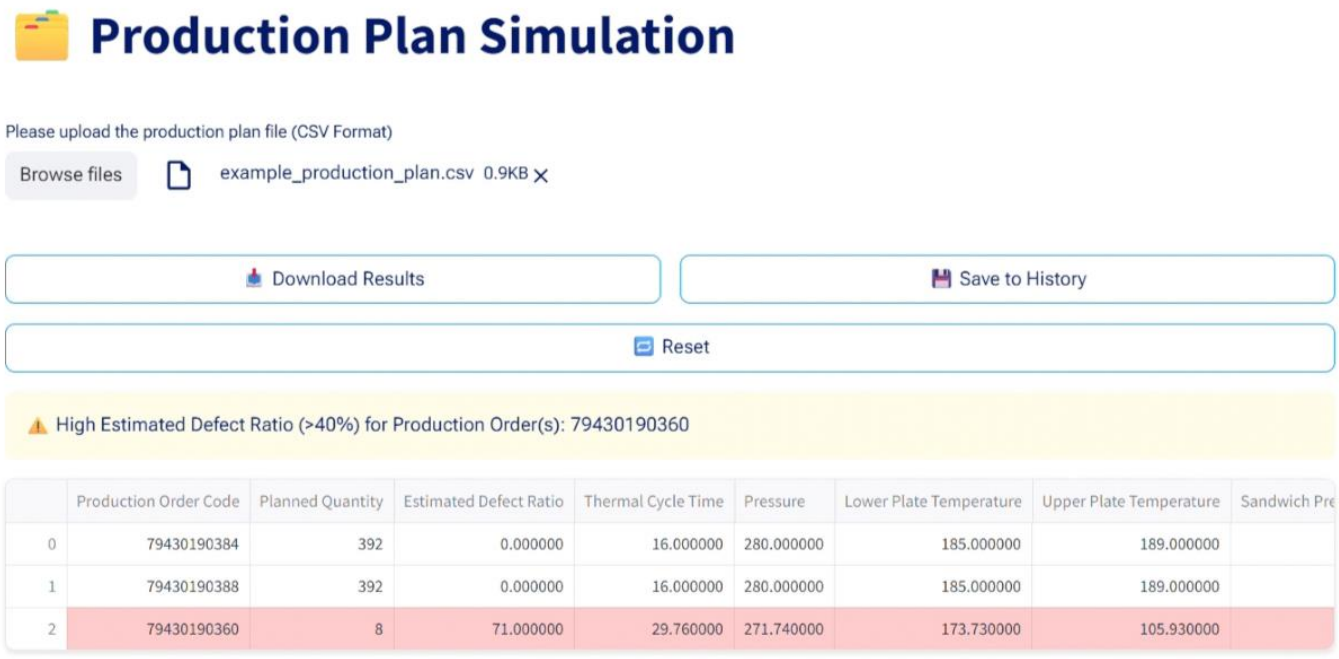


Figure 40: DAT#4 dashboard for the simulation of multiple production orders

4.4.2.6 Quality assessment module for AI model performance monitoring – DAT#5

DAT#5 is responsible for maintaining model performance and enabling continuous improvement through an online monitoring function. This process ensures the system's predictions remain accurate and trustworthy in the dynamic production environment (see Figure 41). The monitoring loop operates as follows:

- Track Predictions: The system continuously tracks the number of defect predictions made during a specific production order.
- Operator Feedback: Once the production order is complete, an operator inputs the actual number of defective panels observed.
- Calculate Error: The system compares the predicted defect ratio against the real defect ratio. It calculates the absolute difference between these two ratios, known as the defect ratio error.
- Retraining: This error value is presented to operators to assess model performance. A low and stable error fosters operator trust, while a significant deviation signals that the prediction model needs to be retrained to adapt to new conditions.

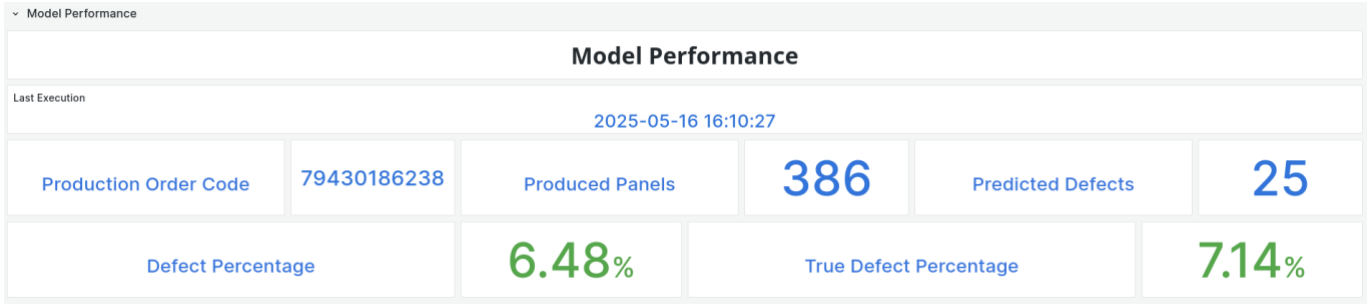


Figure 41: DAT#5 dashboard, illustrating the prediction model performance

4.4.2.7 Quality assessment module for predicting the horizontal distribution of product parameters based on IR images – DAT#6

DAT#6 is a bottle segmentation service based on a semantic segmentation model; it is a generalisation of the segmentation implemented for the NDI9a, running under the TensorFlow framework. The DAT#6 receives thermal images, and the Deep Learning model outlines the contour of a bottle, cropping the area that best fits its shape.

The model is implemented and has been tested, as can be seen in the pipeline generated with the Configurator in Figure 42.

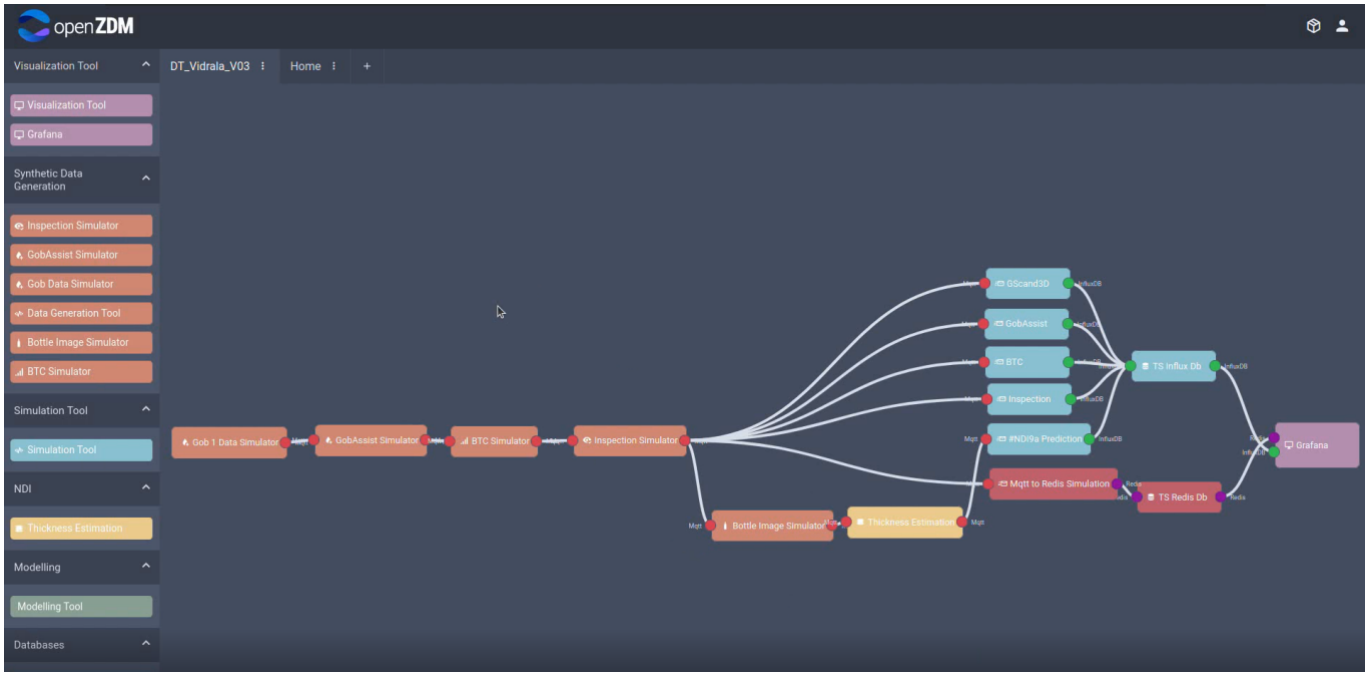


Figure 42: DAT#6 docker available as service in the Configurator

4.4.2.8 Quality assessment module for process parameters reconfiguration – DAT#7

DAT#7 has been developed to correlate one of the thickness measurements taken on glass bottles at the cold end with a set of parameters recorded during gob formation (see Figure 43). The purpose of the service is to assess gob quality based on its potential impact on thickness deviations in the bottles. With the information provided by DAT#7 will be included as optimal ranges of operation and supervised by a set of alarms.

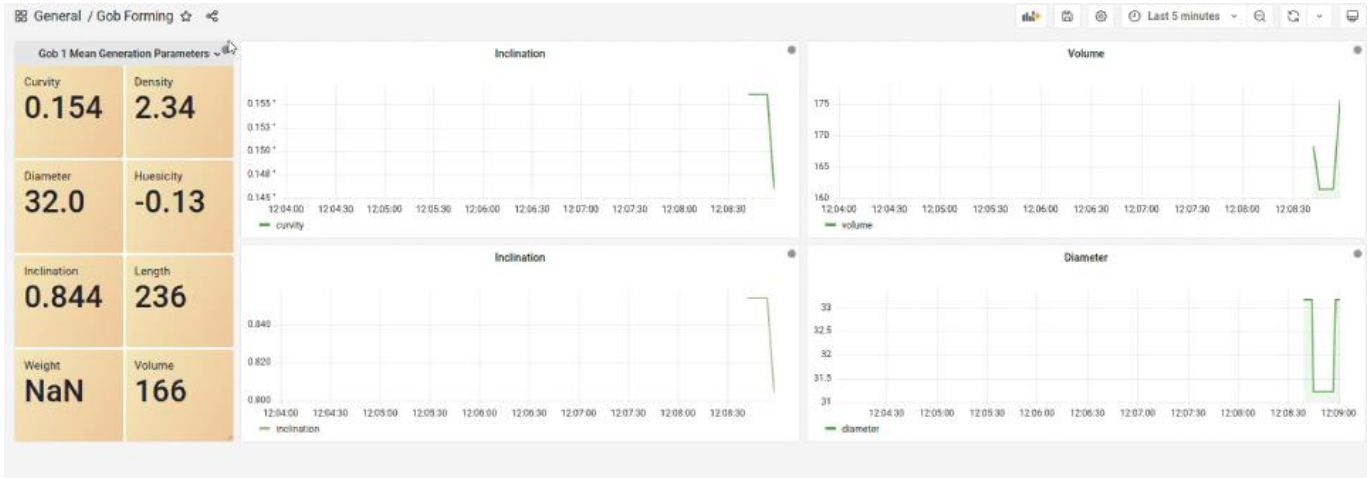


Figure 43: Monitoring of Gob parameters by NDI9b and alarm monitoring by DAT#7

4.4.2.9 Quality assessment module that predicts defects in automotive parts assembly – DAT#8

DAT#8 predicts issues in the front-end and rear-end gap and flush parameters at the end of the Body Shop process, helping with early deviation detection and ensuring vehicle quality. In Figure 44, it is possible to consult the expected problems that a measurement at one station will cause at the next station (either still on the Body Shop or on the Assembly Line). It lists the problem points at the reading station and the points that will be impacted by this measurement problem detected at the next station, showing the name of the point and the condition it should be in at the end of the process.

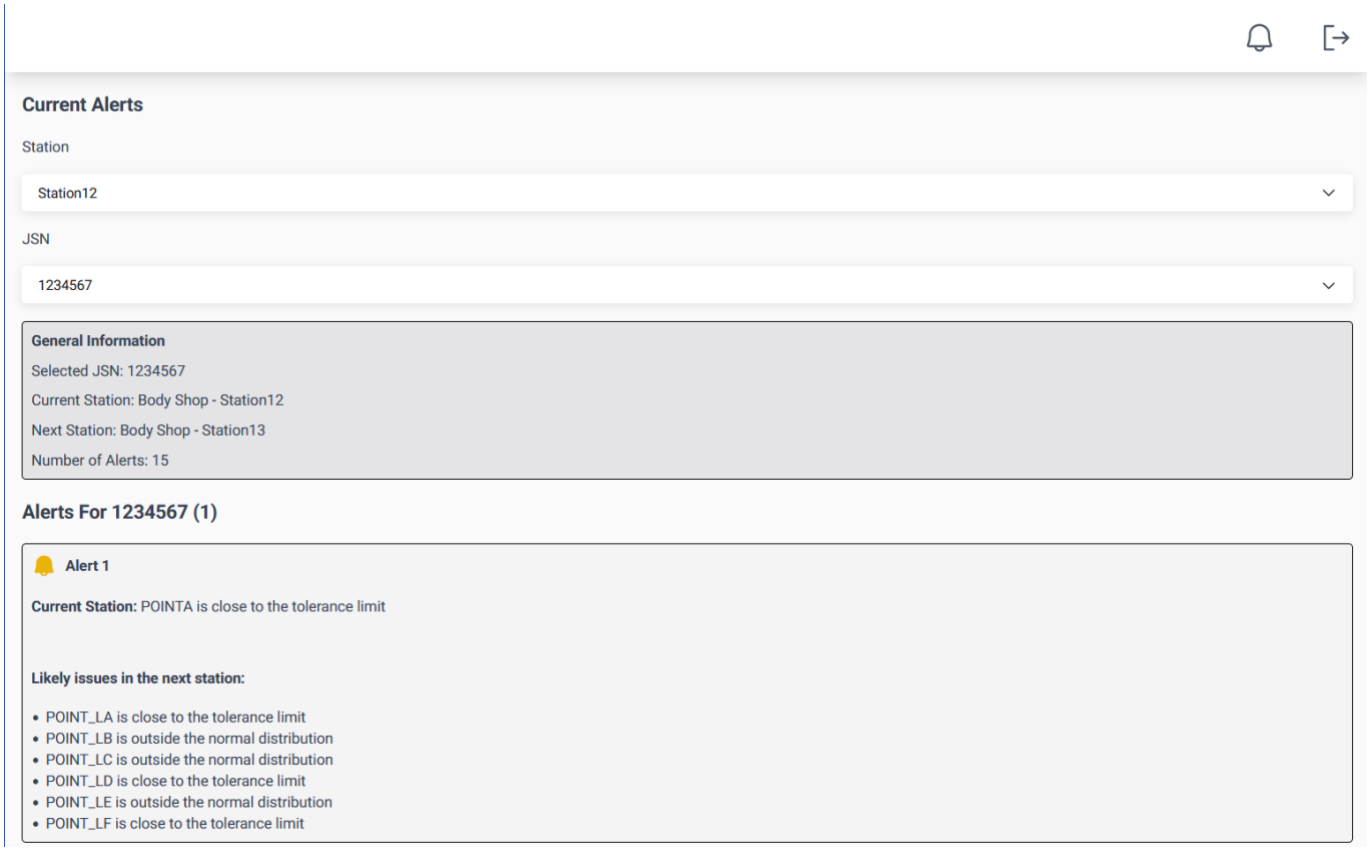


Figure 44: DAT#8 Alert Interface

4.4.2.10 Quality assessment module for predicting points of failure – DAT#9

DAT#9 predicts which checking points on the assembly line may fail based on data from the final Body Shop Station (e.g., DTB2) enabling early detection of potential issues and supporting overall quality assurance. In Figure 45, it is possible to consult the forecast data for a vehicle that was measured at a given station and how it will behave at the next station, thus allowing control and understanding of how one production line is directly impacting another. After the measurement is confirmed at the next station, the user has the option to compare the two measurements and check what was predicted and what was actually measured, thus also providing an idea of the accuracy of the forecast and the adaptation of the model according to the time and evolution of the measurements.

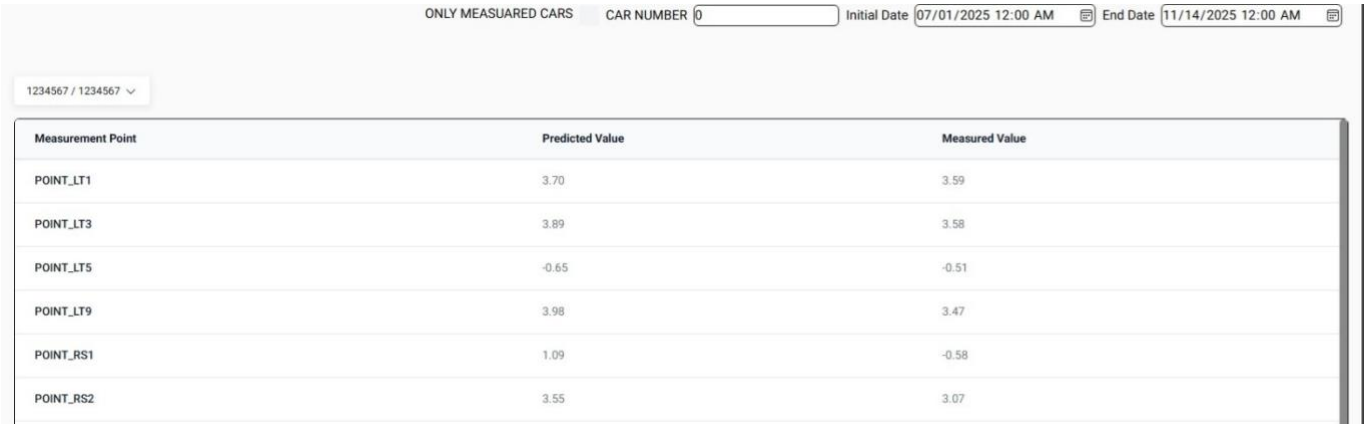


Figure 45: DAT#9 Predictive Model Interface

4.4.2.11 Quality assessment module for root cause analysis – DAT#10

DAT#10 is responsible for defect detection, classification, and cause–and–consequence evaluation within the monitoring tool, leveraging structured assessments of product and measurement-system quality based on collected data. In Figure 46, the presentation of the diagnostic tool data is shown, allowing the user to verify whether the car has passed or failed after analysis and to identify the problematic points, along with potential actions to address

them in the process. The user can consult the underlying causes of the issues, grouped by common problem types for each point, as well as understand their consequences on the final product and the recommended corrective measures. The car's score, calculated from the latest measurements, can also be reviewed, enabling more rigorous control and early evaluation of process trends.

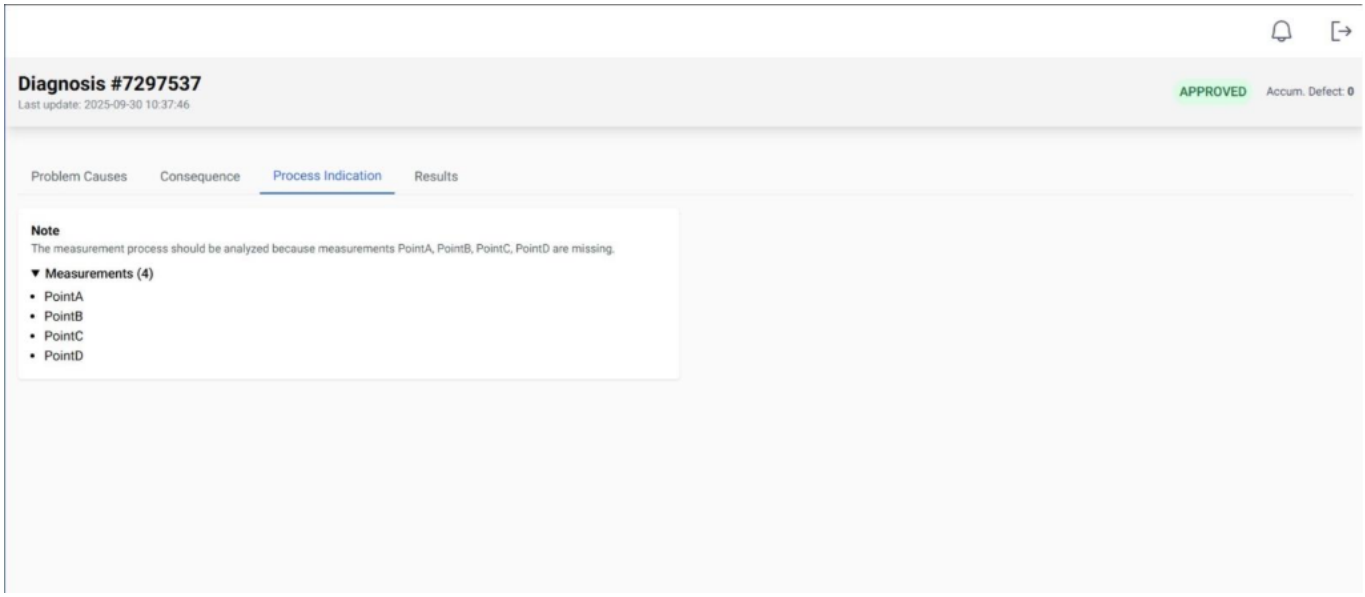


Figure 46 DAT#10 Diagnosis Interface

4.4.2.12 Quality assessment modules for product characteristics monitoring – DAT#11

DAT#11 is responsible for continuously assessing incoming data to detect concept drifts, such as changes in measurement patterns or production deviations caused by equipment wear or environmental shifts. For instance, Figure 47 shows the real-time visualisation interface of the production line, where the user can monitor the most relevant points selected for closer inspection. This enables the user to receive alerts for these points and zoom in on a specific station when necessary to understand what is happening in real time. In this way, the user can quickly identify critical points that may be producing inconsistent measurements, indicating a potential failure in the measurement process.

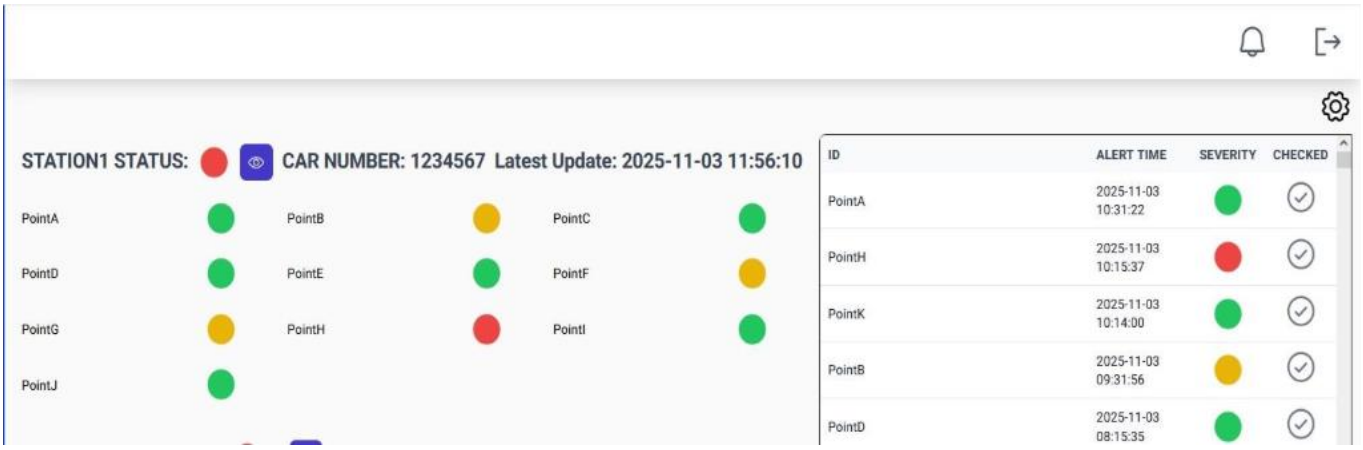


Figure 47: DAT#11 Real-time monitoring interface

4.4.2.13 Quality assessment module for production KPIs monitoring – DAT#12

This quality assessment module is capable of calculating core descriptive statistics of manufacturing systems in relation to their operation and quality of products, such as MTBF, Cycle Time, number of defects in a given time range and others. Such metrics are calculated on top of the digital twin, within the Node-RED instance and are visualised using an off-the-shelf component for graphs and charts, Grafana. This visualisation is illustrated in Figure 48 and demonstrated in the video of 4.4.

Figure 48: DAT#12 Quality assessment module responsible for visualising core production KPIs

4.4.2.14 Quality assessment module for production parameter reconfiguration – DAT#13

This quality assessment module adopts a similar approach to DAT#1; however, it targets the production of EV battery cells instead of large metal bars. Similar to DAT#1, this quality assessment module exposes a user interface that visualises the current setting of the parameter alongside the recommended parameter that the operators can decide to adopt or reject. The user interface is built on the same principles as DAT#1; thus, it is illustrated in section 4.3.2.2. Lastly, on top of DAT#13 an explainable AI layer can be connected, similarly to DAT#0, to enable better understanding on the root-cause of defects for human operators. In particular, the layer highlights the importance of different welding parameters in the detected EV battery welding defect. This in term enables real-time quality inspection and thus, control.

4.4.3 Decision Support Tool

The demonstration of the DST will follow the scenario of the robotic arm, used in the DTT configuration and demonstration. Initially, the user assigns a new name and description to the scenario that will be simulated. This is illustrated in Figure 49.

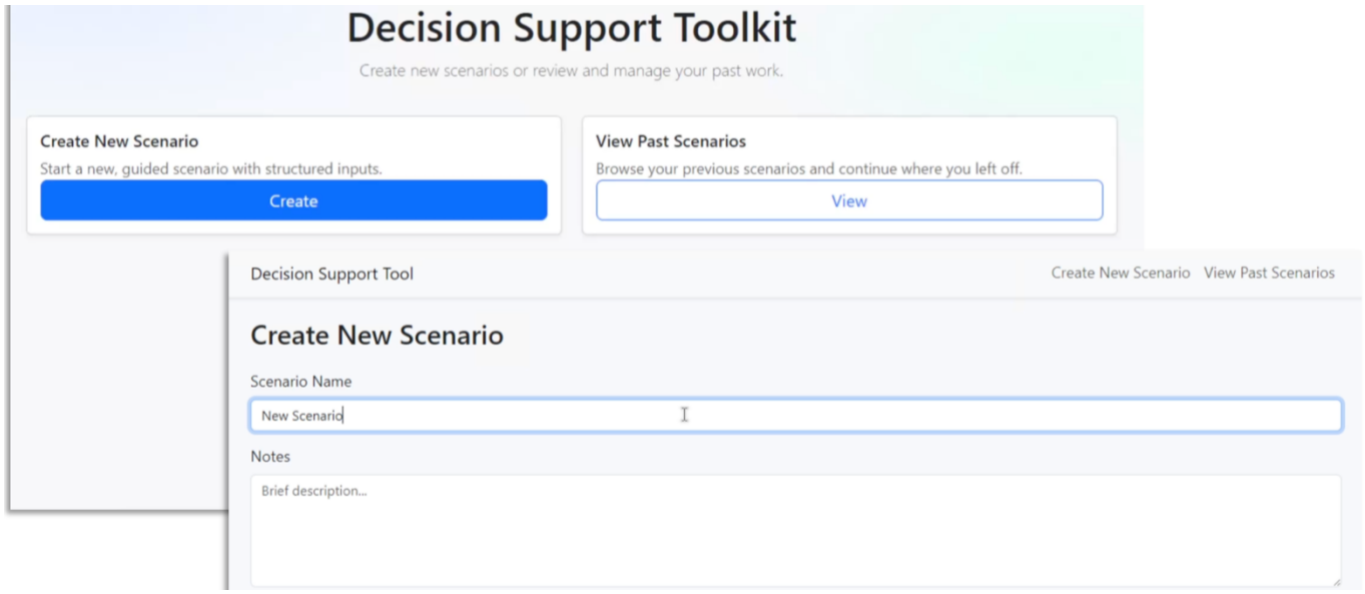


Figure 49: Initialisation of a new scenario in the DST

Following the initialisation, the time horizon of the scenario is selected. The user of the DST has the capability to select between a preconfigured set of months (3, 6, 12) or a custom number of months, as low as 1 month. Following the time horizon definition, the user is requested to add the minimum and maximum ranges of the simulation for each input parameter to be included in the specific scenario, as well as cost/benefit parameters that should be optimised. In the current scenario, the user desires to minimise by at least 10% (-10% improvement as it's a reduction) the percentage of defective products produced by the robotic arm. This is illustrated in Figure 50.

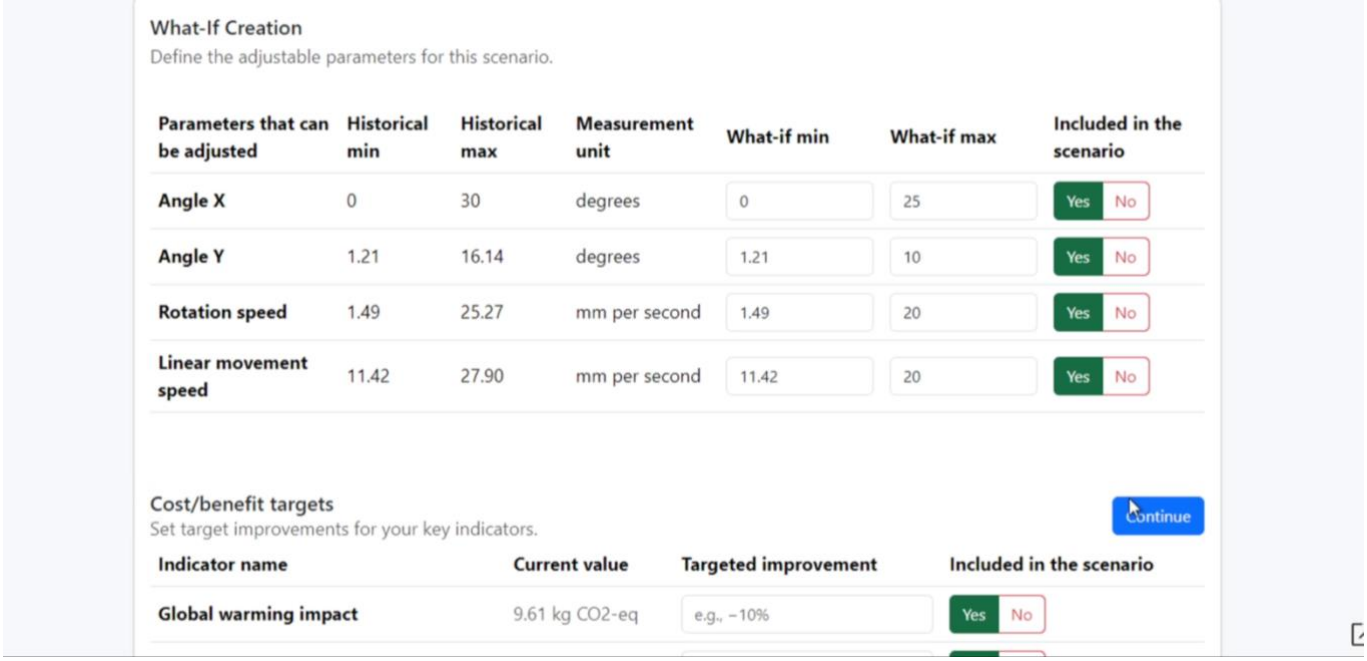


Figure 50: Configuration of time horizon, inputs & cost/benefit indicators in the DST

Lastly, the user configures the optimisation approach that the DST will adopt to try and achieve the desired improvement in the percentage of defective products produced by the robotic arm. The DST allows the use of a reinforcement learning algorithm, a multi-heuristic algorithm and a genetic algorithm. In the current scenario, the multi-heuristic algorithm is selected, given its low computation time when compared to the other two optimisation algorithms. Upon optimisation, the results are presented to the user together with the per-timestamp configurations to achieve the desired improvement by the end of the time horizon. This is illustrated in Figure 51.

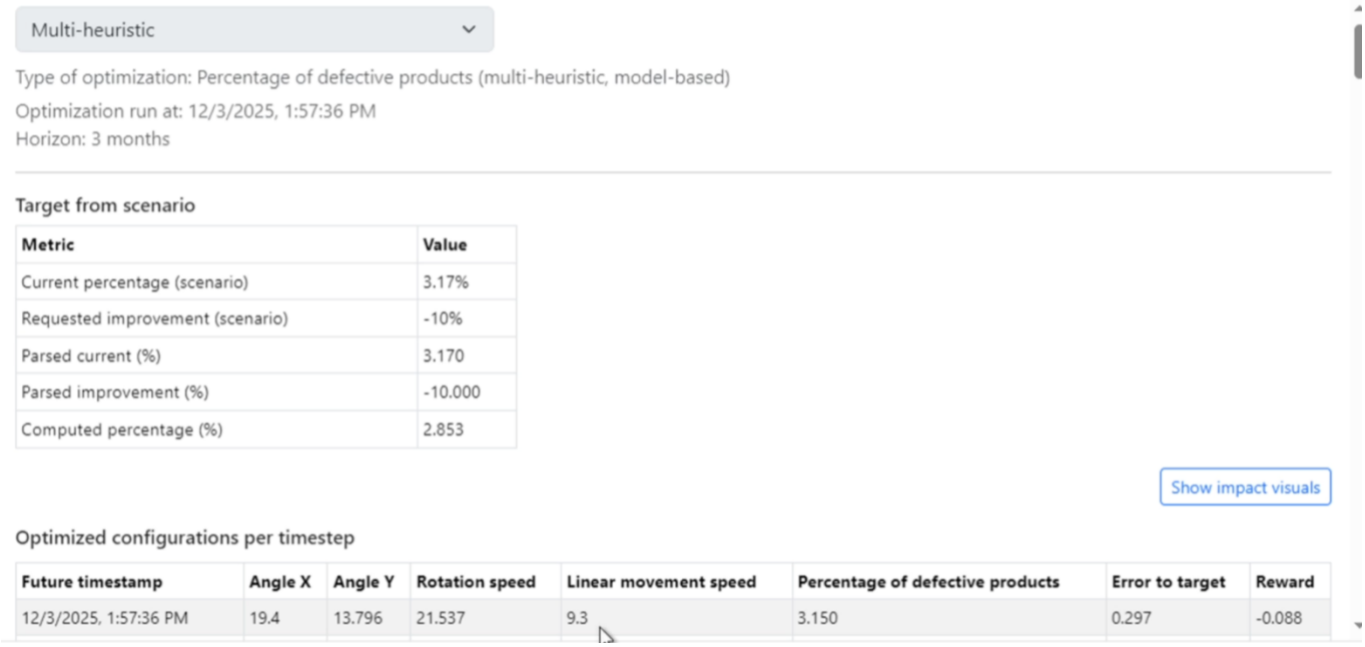


Figure 51: Optimization results viewed through the DST

4.5 Asset Administration Shell type 3

The Asset Administration Shell type 3 functionality that is implemented in openZDM allows the AASs to work together to achieve various goals. The implementation was based on VDI/VDE 2193 [9]. In the project, this functionality is used for two purposes:

- Copy data from one AAS submodel element to a submodel element of another AAS through message exchange.
- Create a new AAS based on a template AAS.

These two capabilities have been integrated in the openZDM platform and allow the generation of instances of AAS products based on an AAS product template and copy data into that product instance from NDI or other shop floor asset AASs. Figure 52 shows an instance of a product created using AAS type 3.

Figure 52: A product instance generated with AAS type 3

4.6 Video

A walkthrough of the final version of the integrated platform and apps is provided below. The individual steps are provided hereafter:

- The configuration section demonstrates the following components:
  - openZDM Platform where a user configures the users, notifications, and available apps to display,
  - DTT where the user creates a new digital twin,
  - Quality assessment module training application where a new DAT is created,
  - The configurator where a DAT is deployed.
- Following the configuration section, operation showcases the runtime of the openZDM solution:
  - Historical data extraction through the AAS, in the openZDM platform,
  - The user uses DAT#0 to consume dimensional predictions,
  - The user uses DAT#1/13 to consume production parameter recommendations,
  - The user uses DAT#2 to consume information on the current environmental performance of the system,
  - The user uses DAT#3 to observe real-time production parameters,
  - The user uses DAT#5 to monitor the performance of an available AI-based analytical module,
  - The user uses DAT#4 to simulate defects in production and get a suggestion on critical parameters' settings,
  - The user uses DAT#6 to consume predictions on production parameters,
  - The user uses DAT#7 to reconfigure production parameters,
  - The user uses DAT#8 and DAT#9 to identify preemptively deviations in the quality of produced products,
  - The user uses DAT#10 to understand the root-cause of defect generation,

- The user uses DAT#11 to understand the correlation between measurement parameters,
- The user uses DAT#12 to view the current and past state of critical production KPIs,
- The user uses the DST to define and analyse alternative what-if scenarios,
- The user uses the configurator to monitor the status of deployed components.

A video showing the corresponding demonstrator is available, upon request, through the following link:

LINK

5 Security, scalability, replicability, and performance evaluation

In order to address the security aspect of the platform, a set of guidelines has been defined at the beginning of the platform development process, following OWASP recommendations [10]. These provisions include:

1. Provisions for maximising the effectiveness of access control (authorisation).
2. Provisions for avoiding cryptographic failures, such as using deprecated cryptographic functions.
3. Provisions to avoid Injection attacks in the platform (e.g. avoid using user input directly into queries).
4. Adoption of secure design patterns.
5. Provisions for avoiding security misconfigurations (e.g. use different credentials for production and testing environments).
6. Using up-to-date components and monitoring security updates.
7. Provisions for using secure passwords.
8. Logging access events and authentication failures in the platform.

In addition to the security provisions user interface and backend APIs are protected using the authentication and authorisation mechanisms described in section 1.3.

By design, the platform has been developed as a set of services running independently from each other. While some services rely on others for providing their functionality, all services can be deployed individually. This allows for vertical and horizontal scalability.

Platform replicability has been verified by applying different configurations of the openZDM platform targeting use cases in the domains of battery production, wood panel production, the metal industry, automotive industry, and the bottle industry.

In terms of performance, the evaluations have been carried out. The openZDM project defines specific KPIs for WP4 under its SMART objectives. These are presented in Table 2 along with their measuring results.

Table 2: openZDM platform evaluation

#	KPI	Target	Achieved	Notes
1	The platform will support over the cloud and hybrid deployment	Cloud, Hybrid	Yes	Platform deployments include: 1. Fully cloud: all components are in the cloud. 2. Hybrid: Platform software hosted in the cloud; NDI software hosted on the shop floor close to the NDI hardware. 3. Local: The platform is hosted in the local infrastructure of the pilot.
2	The platform should allow for multiple users with different workflows, preserving data isolation	8 users	Yes	A specially designed test was carried and verified that 8 users can be connected to the platform in parallel with different workflows and data isolation.
3	Data turnover times: The time from the data creation in the sensor/NDI connected to the platform until the storage in the physical layer of the platform	Under 4.9 seconds	Yes	1. Data turnover for small payloads on average less than 1 second. 2. Data turnover for large payloads (2MB) on average less than 1 second.
4	Digital twins to be successfully used in multiple use cases	At least 3 cases	Yes	Digital twins are used in all use cases.

5	Digital twins update with live data from at least 3 sources	Update in less than 1.5 sec	Yes	Less than 0.5 second achieved.
6	Number of data-driven tools to be developed and validated in each use case	At least 1 tool per use case	Yes	Each use case has at least one data-driven tool developed.

6 Conclusions

This deliverable is a demonstrator of the final developments of the openZDM platform and its applications (i.e. the Digital Twin Toolset, the data-driven Quality Assessment Modules (DATs), the Quality Assessment Module Model Training App, the Decision Support Tool), including its services and integration.

All developed applications are complex (may include several software components, databases, etc.) and may work in combination with each other or standalone. In addition, as detailed in section 3, several configurations are required in order to make the platform deliver its functionalities to the end users. An effort has been made to simplify the user interfaces as much as possible; however, some complexity still exists due to the nature of some of the applications. For example, the Digital Twin Toolset offers three modelling approaches (i.e. data-driven, physics-based, and a combination of the two), thus creating a physics-based model or a PINN model requires substantial knowledge of the underlying system's behaviour described in differential equations. In addition to simplifying the user interfaces, to further enhance accessibility, the platform allows for configuration of views per user, meaning that specific users may have a reduced amount of information, accessing only very relevant information for their work, whereas others may have more complex views with more information. Moreover, multi-language support was added to support users who are not native English speakers. Lastly, an inclusive design approach was used during the development of the openZDM platform, avoiding gender bias in colour selection, visuals, and gender assumptions. Gender-inclusive language is used.

The platform heavily relies on Asset Administration Shells, a concept still under heavy development, meaning a lack of existing implementations and documentation (especially for AAS type 3), thus adding to the complexity of the development effort of the platform. Limitations, either due to the AAS specification or the existing implementations (e.g. lack of searching API for searching in the AAS repository by keyword), led to some workarounds to access data (e.g. accessing the AAS database directly instead of the AAS API). Moreover, while all data are represented using AAS, the AAS concept and standards do not provide mechanisms/approaches for storing historical data of the models.

In addition, in the development of DATs, a pivotal challenge has been data collection. Given the high diversity in legacy systems used by industrial end-users, data collection has been a key challenging aspect in the validation of the developed methodologies. Nonetheless, the use of a single middleware (openZDM AAS Middleware) has streamlined data access during the deployment of developed tools. Moreover, it should be noted that the concept of AAS proved valuable in the integration of sensor equipment, such as the NDIs developed in the project, by abstracting the NDI hardware/software through the AAS model and API.

For a wider industrial adoption, some actions are important to be undertaken. These include improving the robustness of the existing AAS implementation to avoid performance bottlenecks when a significantly high number of AASs is present in the repository. In addition, tools such as the DTT and the DST need to be simplified, and their UIs need to become more streamlined to be accessible by a wider range of employees and not be limited only to experienced process engineers.

References

[1] ZVEI, "Reference Architecture Model Industrie 4.0 (RAMI4.0)", Jul. 2015. [Online]. Available: https://www.zvei.org/fileadmin/user_upload/Presse_und_Medien/Publikationen/2016/januar/GMA_Status_Report__Reference_Archtitecture_Model_Industrie_4.0__RAMI_4.0_/GMA-Status-Report-RAMI-40-July-2015.pdf. [Accessed Sept. 26, 2022].

[2] K. Alexopoulos, K. Sipsas, E. Xanthakis, S. Makris & D. Mourtzis, "An industrial Internet of things based platform for context-aware information services in manufacturing", International Journal of Computer Integrated Manufacturing, 31:11, 1111-1123, DOI: 10.1080/0951192X.2018.1500716, 2018

[3] O. Lázaro, et al., “Big Data-Driven Industry 4.0 Service Engineering Large-Scale Trials: The Boost 4.0 Experience”, In: Curry, E., Auer, S., Berre, A.J., Metzger, A., Perez, M.S., Zillner, S. (eds) Technologies and Applications for Big Data Value . Springer, Cham. https://doi.org/10.1007/978-3-030-78307-5_17, 2022

[4] The openZDM Platform [Online]. Available: https://www.openzdm.eu/wp-content/uploads/2024/06/Technical-Articles_The-openZDM-Platform.pdf

[5] Platform Industrie 4.0, “Asset Administration Shell Reading Guide”, 2022. [Online]. Available: https://www.plattform-i40.de/IP/Redaktion/DE/Downloads/Publikation/AAS-ReadingGuide_202201.pdf?__blob=publicationFile&v=1

[6] Platform Industrie 4.0, “What is the Asset Administration Shell from a technical perspective?”, 2021. [Online]. Available: https://www.plattform-i40.de/IP/Redaktion/EN/Downloads/Publikation/2021_What-is-the-AAS.pdf?__blob=publicationFile&v=1

[7] International Digital Twin Association submodel repository, 2025. [Online]. Available: https://industrialdigitaltwin.org/en/content-hub/submodels

[8] OAuth2.0 Protocol. 2025. [Online]. Available: https://oauth.net/2/

[9] VDI/VDE 2193 Sheet 1: Language for I4.0 components, VDI, Düsseldorf, 2019VDI/VDE 2632 Blatt 1 – Machine Vision – Basic terms, and definitions (2023).

[10]OWASP Top 10, 2021. [Online]. https://owasp.org/Top10/A00_2021_Introduction/


openZDM