




D3.3 DATA FUSION METHODS AND DIGITAL TWIN BASED IOT DATA PROCESSING PIPELINES

Manuel Imperiale; MFL

@AshvinH2020 
ASHVIN H2020 Project 
www.ashvin.eu 



ASHVIN has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No 958161. This document reflects only the author's view and the Commission is not responsible for any use that may be made of the information it contains.

Project Title	Assistants for Healthy, Safe, and Productive Virtual Construction Design, Operation & Maintenance using a Digital Twin
Project Acronym	ASHVIN
Grant Agreement No	958161
Instrument	Research & Innovation Action
Topic	LC-EEB-08-2020 - Digital Building Twins
Start Date of Project	1st October 2020
Duration of Project	36 Months

Name of the deliverable	Data fusion methods and digital twin based IoT data processing pipelines
Number of the deliverable	D3.3
Related WP number and name	WP 3 Data fusion for real-time construction monitoring
Related task number and name	T3.2 IoT data processing / T3.4 Data fusion
Deliverable dissemination level	PU
Deliverable due date	31-08-22
Deliverable submission date	31-09-22
Task leader/Main author	Manuel Imperiale (MFL)
Contributing partners	Athina Tsanousa (CERTH), Konstantinos Ioannidis (CERTH), Stefanos Vrochidis (CERTH), Manuel Jungmann (TUB)
Reviewer(s)	Timo Hartmann (TUB), Jason Pridmore (EUR)

ABSTRACT

This deliverable describes the development of communication protocols and security procedures for IoT data processing and secure real-time data stream pipelines from physical data sources (devices, sensors, and actuators). This deliverable also covers the development of data fusion algorithms necessary for collecting, processing combining and fusion of data and information from multiple sources and services.

KEYWORDS

Digital twin, Internet of Things, IoT protocols, Data fusion, AI, Machine Learning

REVISIONS

Version	Submission date	Comments	Author
V0.1			
V0.2			
V0.3	22/9/2022	Revised after internal review	All
.....			
.....			

DISCLAIMER

This document is provided with no warranties whatsoever, including any warranty of merchantability, non-infringement, fitness for any particular purpose, or any other warranty with respect to any information, result, proposal, specification or sample contained or referred to herein. Any liability, including liability for infringement of any proprietary rights, regarding the use of this document or any information contained herein is disclaimed. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted by or in connection with this document. This document is subject to change without notice. ASHVIN has been financed with support from the European Commission. This document reflects only the view of the author(s) and the European Commission cannot be held responsible for any use which may be made of the information contained.

ACRONYMS & DEFINITIONS

IoT	The Internet of things
IIoT	The industrial internet of things
CERTH	Centre of Research & Technology - Hellas
DTT	DigitalTwin Technology GmbH
ReBAC	Relationship-based access control
SenML	Simple information model for sensor measurements and device parameters
TLS	Transport Layer Security
SSL	Secure Sockets Layer
CLI	A command-line interface
SSO	Single sign-on
CRUD	Create, read, update and delete, functions
AWS	Amazon Web Services
LoRaWAN	Low-power wide-area network
TUB	Technical University of Berlin
UPC	Polytechnic University of Catalonia
GPS	Global Positioning System
KNN	K-Nearest Neighbors
ML	Machine Learning
SVM	Support Vector Machines
PUC	Pilot Use Case
IMU	Inertial Measurement Unit
MAE	Mean Absolute Error
MSE	Mean Squared Error

ASHVIN PROJECT

ASHVIN aims at enabling the European construction industry to significantly improve its productivity, while reducing cost and ensuring absolutely safe work conditions, by providing a proposal for a European wide digital twin standard, an open source digital twin platform integrating IoT and image technologies, and a set of tools and demonstrated procedures to apply the platform and the standard proven to guarantee specified productivity, cost, and safety improvements. The envisioned platform will provide a digital representation of the construction product at hand and allow to collect real-time digital data before, during, and after production of the product to continuously monitor changes in the environment and within the production process. Based on the platform, ASHVIN will develop and demonstrate applications that use the digital twin data. These applications will allow it to fully leverage the potential of the IoT based digital twin platform to reach the expected impacts (better scheduling forecast by 20%; better allocation of resources and optimization of equipment usage; reduced number of accidents; reduction of construction projects). The ASHVIN solutions will overcome worker protection and privacy issues that come with the tracking of construction activities, provide means to fuse video data and sensor data, integrate geo-monitoring data, provide multi-physics simulation methods for digital representing the behavior of a product (not only its shape), provide evidence based engineering methods to design for productivity and safety, provide 4D simulation and visualization methods of construction processes, and develop a lean planning process supported by real-time data. All innovations will be demonstrated on real-world construction projects across Europe. The ASHVIN consortium combines strong R&I players from 9 EU member states with strong expertise in construction and engineering management, digital twin technology, IoT, and data security / privacy.

TABLE OF CONTENTS

1. INTRODUCTION	8
1.1 Background	9
1.2 Purpose of the document and audience	9
1.3 Outline of the report	9
2. DIGITAL TWIN FOR CONSTRUCTION	10
3. IOT DATA PROCESSING PIPELINES	12
3.1 Ashvin Platform and IoT data processing pipelines	12
3.2 Goals and objectives of IoT data processing pipelines	13
3.3 Our Advancements in IoT Data Processing	15
3.3.1 Secure certificate storage and creation with Vault	15
3.3.2 Single sign-on and user management with Keycloak	16
3.3.3 Open weather data gathering, processing and aggregating with Telegraf	19
3.3.4 User and thing groups authorization policy management with Ory Keto	21
3.3.5 DigitalOcean to AWS Kubernetes migration and automated DB management with AWS and Velero	23
3.3.6 Indoor environmental data gathering, and aggregation with ChirpStack LoRaWAN Network	26
4. DATA FUSION METHODS	29
4.1 GOALS AND OBJECTIVES	29
4.2 Activity recognition and occupancy detection	30
4.2.1 Occupancy detection through environmental sensors	30
i. Related work	30
ii. Implementation	30
4.2.2 Activity recognition for duration extraction	34
4.2.3 Fusion of sensors for water levels correlation	39
5. CONCLUSIONS	42
6. REFERENCES	43

INDEX OF FIGURES

Figure 1 Ashvin Platform (AP)	16
Figure 2 Kubernetes cluster with Vault hosted in Kubernetes pod	18
Figure 3 Keycloak login screen	18
Figure 4 Mianflux Dashboard screen	18
Figure 5 Keycloak admin panel	18

Figure 6 Weather data in JSON in Mainflux UI	20
Figure 7 Mainflux UI groups	21
Figure 8 Mainflux UI users	22
Figure 9 Mainflux UI users and users details	22
Figure 10 AWS Kubernetes cluster admin panel	25
Figure 11 AWS Kubernetes cluster admin	25
Figure 12 AWS Kubernetes cluster admin panel	28
Figure 13 AWS managed DB admin panel	28
Figure 14. The Things Network LoRaWAN Network Server admin panel	27
Figure 15. The Things Network LoRaWAN Network Server admin panel	28
Figure 16. ChirpStack admin panel	28
Figure 17. Final Data format (PUC 2)	32
Figure 18. Fluctuation of mean humidity value with the change of temperature (PUC 2).....	33
Figure 19: Correlation of sensor values (PUC 2)	34
Figure 20: Feature importance (PUC 2)	35
Figure 21: Missing/NaN values (PUC 2)	35
Figure 22. Methodology pipeline	34
Figure 23 Picture of the device mounted on the crane.....	35
Figure 24. Operations (activities) performed by the crane	35
Figure 25. Positions of fibre optic sensors of PUC10 Quay wall	40
Figure 26. marks the spot of the locations that were selected (PUC 10).....	41
Figure 27. Distribution of PUC10 online data after pre-processing	40
Figure 28. Distribution of PUC10 data after pre-processing	40
Figure 29. Processing Pipeline for PUC 10	41

INDEX OF TABLES

Table 1: Classifier accuracy for 10-fold cross validation	36
Table 2: Confusion matrix for Random Forest	37
Table 3: Precision and Recall for the Random Forest classifier	37
Table 4: Resulting operation durations.....	38
Table 5. Correlation results	41

1. INTRODUCTION

This deliverable report is part of Work Package (WP) 3 of the ASHVIN project titled "Data fusion for real-time construction monitoring".

The objective of WP3 is a development of novel algorithms that enable the extraction of features from the real world for mapping to simulated reality. As the important phases for that purpose, Tasks, 3.2 IoT Data Processing and 3.4 Data fusion, which these deliverable addresses, are designed to provide secure real-time data stream pipelines from physical data sources (T3.2) and the development of the necessary algorithms for collecting, processing, and fusing data from a variety of sources and services (T3.4).

1.1 Background

Productivity, costs and resource efficiency, as well as safety in the construction industry, are still excessively low compared to other industries. Two main reasons for that are vertical and horizontal fragmentation between all stakeholders within and across the design/engineering, construction, and maintenance of buildings and infrastructure. The other is the inherent disposition of construction itself related to that much of the work involves the delivery of one-of-a-kind products that must be manufactured at the location where the products will be used.

In the past decades, the rapid development of digital technologies enabled significant improvements in day-to-day operations and processes in many industries. However, the construction industry is still slow in the adoption of advanced technologies.

The Internet of things, digital twin, advanced digital scanning and imagery, machine learning, and AI have a powerful capacity to enable a holistic approach that will address fragmentation by integrating information and knowledge from the design and engineering phases to construction and further on to maintenance stages.

Current initiatives and utilisation of the above-mentioned technologies did not advance much further from collecting large amounts of data from construction activities. During the planning process, however, it is impossible to take into account the significant relationships between various construction processes, construction design, and environmental data.

ASHVIN project is conceived to address these inherent challenges with a holistic approach that deploys digital technologies through all stages of the construction development lifecycle.

This will be achieved by provisioning the IoT-based digital twin platform, data fusion methods, AI analytics, visualisation, and specific applications and processes that will collect, use and integrate vast amounts of different data coming from the entire design /engineering and construction and maintenance process.

Within the task T3.2 of the ASHVIN project, the focus is to develop and provide secure real-time data stream pipelines from physical data sources devices, sensors, actuators and virtual data sources in the cloud or on-premise applications that gather or generate data - to the IoT platform. The latter is meant to serve as a central place for secure data storage and retrieval and, thus, to cater to the data needs of the Digital Twin platform and various design and engineering applications.

While T3.4 creates algorithms and methods required to gather, analyze, and integrate information from diverse sources and services. Sources including various sensors (humidity, dust, smoke, water level, etc) are merged initially, allowing for the discovery of possible correlations between them. Pre-processing, fusion and machine learning algorithms are then employed to derive results and conclusions. During this task, data will be extracted from the IoT platform, pre-processing techniques will be applied to improve the quality of data and suitable fusion methods will be adopted, to maximise the information extracted from data.

Summarising, data fusion has been applied for object activity recognition (PUC6) from heterogeneous sensors attached to a crane, occupancy detection from non-intrusive environmental sensors (PUC2 and 5), and water level correlation analysis from sensors installed on a bridge and from sensor data collected from a web site.

1.2 Purpose of the document and audience

The intended audience for this report is software developers who are planning to provide secure IoT data collection and data fusion methods from multiple sources and develop digital twin solutions for construction sites, whether they are general digital twin platforms or particular applications for utilising IoT data.

In addition, the report is intended for IT managers and R&I specialists who want to better understand secure IoT data acquisition and plan to implement specific digital twin solutions on construction projects.

1.3 Outline of the report

This document is structured into 4 sections. The first section presents the background and the objective of the deliverable report. Section 2 explains in more detail the problems of the construction industry and the potential of digital technologies - in particular, IoT and digital twin - for solving them. In section 3 process of development of secure real-time data stream pipelines is presented, while section 4 elaborates on the creation of data fusion mechanisms, deployed to combine diverse types of data collected by different technologies.

2. DIGITAL TWIN FOR CONSTRUCTION

Despite technological advancements over the past few decades which enabled significant improvements in day-to-day operations and processes in many industries, the construction industry continues to have very low productivity, efficiency, and safety in comparison to other industries.

One of the most significant causes of these problems is vertical and horizontal fragmentation which could be interpreted as a sort of invisible walls between project stages and between organisations and professions in a project's lifecycle. Limited space for standardisation, mass production, and flexibility is the second reason for the construction industry's primary challenge. This fragmentation creates the circumstances in which the majority of design and engineering is conducted without consideration for the construction processes required to realise the designs. During the planning process, it is impossible to take into account the significant connections between various construction processes, construction design, and environmental data.

The value that digital technologies can provide to the construction industry comes from the integration of a huge amount of data, information, and knowledge from design /engineering phases to construction and further on to maintenance stages. Integration of that kind can be enabled with a wide spectrum of technologies such as the internet of things and its real-time continuous data acquisition, digital twin technology, data fusion methods, AI analytics, and visualisation, with specific applications and processes that can utilise vast amounts of different data, that have the potential to increase productivity, improve resource efficiency, and safety in the construction industry.

Digital twin is the most substantial technology for solving fragmentation in the construction industry. It should be understood as a digital/virtual representation of actual or potential physical objects, assets, devices, systems, and processes. It can provide detailed information, feedback, and insights about its real-world counterparts (on their properties, status, and performance) in real-time, with the purpose of optimising processes, detecting current and potential issues, predicting outcomes, and creating better products.

Utilising the internet of things, sensors, realistic simulation, and advancements in computer vision and machine learning technologies, it provides vast quantities of cumulative measurements of its real-world counterpart as well as digital reproductions and simulations of the asset.

In the current situation, these advanced technologies are not deployed in a manner that will solve the fragmentation and other problems of the construction industry.

The Internet of things is utilized to collect a large amount of data, with little useful information to manage construction work, and furthermore, this data is not available through a central, single point of access which will enable insights into relations between major construction stages and processes. Digital twin initiatives, on the other hand, concentrate on the integration of only very specific streams of data, focusing

primarily on image-based, equipment-based, structural-monitoring-based, or geo-monitoring-based information.

The ASHVIN project is designed to address these inherent challenges with a holistic approach that provides seamless integration of all stakeholders and vertical and horizontal interoperability of data and information within and across design/engineering, construction, and maintenance of buildings and infrastructure.

This will be achieved with the deployment of digital twins that will provide a dynamic digital representation of buildings or infrastructure systems, as well as a representation of all significant processes surrounding them during all stages of the project development lifecycle.

Enabled by an industry-ready IoT platform that provides data from sensors, photogrammetric data, laser, and thermal scans, images and video, digital twin in this context will include an accurate representation of the real-world conditions, simulation of buildings' and infrastructure' multi-physics behaviour and real-time synchronizations between as-designed and as-built models from design /engineering phases to construction and to maintenance phases.

Furthermore, to fully leverage IoT-based digital twins, the ASHVIN project also envisioned the development of machine learning and AI solutions for data processing and analysis, as well as a vast array of design and engineering applications integrated in the central digital twin platform.

With this approach, IoT-based digital twin can synchronise as-designed and as-built models to continuously monitor real-time progress against initial plans, allowing early detection of discrepancies or reactions and flexible planning for changes.

Combining multiple relevant data sources with AI-reinforced data analytics and fusion methods into one decision central instance also enables holistic monitoring of support activities, such as lead time planning for material and pre-fabricated products, discipline hand-off planning, safety hazard identification, optimization of equipment usage and site logistics planning.

3. IOT DATA PROCESSING PIPELINES

3.1 Ashvin Platform and IoT data processing pipelines

The ASHVIN project's holistic strategy described in the previous section is founded on the Ashvin platform (AP) which consists of several technology layers:

- i) Sensing and video/image/monitoring
- ii) Edge computing
- iii) Ashvin IoT platform (AIP) and game-engine digital twin platform (DT)
- iv) AI analytics, and IoT data fusion techniques based on AI data association techniques
- v) Design and engineering applications — Ashvin Toolkit



Figure 1: Ashvin Platform (AP)

The Ashvin platform (AP) presented in Figure 1, relies on the data gathering from multiple sources to make meaningful use of the Digital twin platform (DT) as well as other applications/tools. It means that DT and other tools need to be able to access the gathered data, preferably in a uniform manner and from a central repository. The accessed data can be historical (data gathered in the past) or real-time (data being collected in the present moment).

In the context of the Internet of Things (IoT), data collection and retrieval are regarded as messaging. Essentially, devices and applications exchange messages. Messages contain payload - usually, measurements originating from devices - and metadata - the part which describes the message itself, such as a message timestamp, etc. The Ashvin IoT platform (AIP), which is based on Mainflux open-source messaging middleware and the Angular GUI framework, is the messaging backbone of the entire AP. This means that DT and other AP tools rely on the AIP to collect and access the needed data.

To serve the purpose of the messaging “backbone”, the AIP has to implement communication pipelines that lead from devices to the central data repository and from the central data repository to applications. Furthermore, these pipelines need to be secure in order to prevent unauthorised access and eavesdropping. Finally, the pipelines need a single point of access from the outside and the inside (the DT and tools) of the AP and a uniform way of data retrieval, again from the outside and the inside of the AP.

These three points: pipeline topology, secure access and retrieval of data and a uniform point and way of data access constitute the main goals and objectives of T3.2.

3.2 Goals and objectives of IoT data processing pipelines

The main outcome of IoT data processing is development of communication protocols and security procedures for enabling secure real-time data stream pipelines from physical and virtual data sources to/from the IoT platform. Physical sources of data include primarily devices, sensors, and actuators. Virtual data sources include cloud or on-premise applications that gather or generate data. In order to meet data requirements of the DT and numerous design and engineering applications/tools, we need a central location/repository for safe data storage and retrieval.

Access to stored data is provided in the form of :

1. real time data streams via MQTT subscription.
2. historical data packages via HTTP REST API

where data is fetched from the InfluxDB database.

Concerning the first option, MQTT is an industry standard messaging protocol for the Internet of Things, where devices or applications can send or receive messages to/from communication channels. Channel acts as an open stream of messages operating in real time.

Concerning the second option, the client sends HTTP requests to the server. The server sends back an HTTP response with the requested message data.

Both options rely on the InfluxDB, a time series database, where data points are stored in a sequence, one after another, and are associated with a timestamp.

In addition, a unified possibility for enabling exclusive access to entire systems within the ASHVIN platform and unified management of platform users were required. In this regard, security measures and procedures must be implemented to prevent unwanted and unauthorized data access.

The full blown security IoT system uses certificates, a sort of device and application identity cards (or passports) checked by the central certificate authority (which verifies the authenticity of certificates). For the moment, the procedure for issuing certificates is not entirely user-friendly - the work is being done in making it easy to use -, so simple device and application credentials are currently used, for convenience.

In the context of the AIP, we use X509 client- and server-side Mutual Transport Layer Security (mTLS) authentication. This means that the client has to provide authentication certificates to the server and vice versa (this is what “mutual” stands for). X509 is a standard defining the format of certificates. Briefly put, every communication party exposes a public key and stores a secret key: a message encrypted (locked) with a public key can be only decrypted (unlocked) with a private key. In a nutshell, an mTLS certificate contains a public key and a reference to the certification authority.

The AIP uses the Vault management system (which could be regarded as digital safe) for issuing and storing device and application certificates. We are currently developing the easy and UI based access to the vault.

Once the certificates are exchanged and checked, that is, once we know that communication parties are really the ones who they claim to be, we need to check what kind of access rights the corresponding party has. Relations-based access control (ReBAC) policies are used to control access (access here means the possibility to read/retrieve and write/store messages) to the AIP. ReBAC is essentially a system used to handle authorisation (permissions) based on roles, attributes or other predicates pertaining to the AIP entities (mainly users and things as well as groups of users and things).

The access predicates (relationships), AKA policies, in the context of the AIP, are built on top of the groups of things (which represent devices and application) and users (things and users are the main entities of the AIP). Policy service itself is based on the open-source implementation of Zanzibar, Google's Global Authorization System. The implementation is still in progress, however, fundamental capabilities like user access rights and sharing of things are finished.

Single sign-on (SSO) is added to the AP. SSO is an authentication system that allows a user to log in with single credentials. In the context of the AP, which consists of the AIP and DT, as well as of numerous applications/tools used to process the gathered data, this means that we don't have to provide separate credentials for each authentication demanding component (the AIP, DT and applications/tools) of the AP. This also facilitates the tighter integration of the AP components. SSO itself is built on the Keycloak and OpenID Connect (widely used open source identity management solution).

3.3 Our Advancements in IoT Data Processing

3.3.1 Secure certificate storage and creation with Vault

Mainflux relies on Transport Layer Security (TLS) for the communication party authentication and message encryption.

TLS is an encryption protocol:

- a) the protocol is a set of rules or procedures on how the exchanged data (IoT messages, here) is formatted and interpreted
- b) encryption is a process of transforming a message into secret code to hide the encrypted information.

This means that TLS is a set of rules of the message encryption, transmission and decryption. TLS ensures the client-server authentication, the procedure of determining whether a client or a server is who or what it says it is.

This means that users and things can establish a secure connection over the IoT network. In the AIP terminology, “users” are entities that represent people and organisations, and “things” are entities that represent physical devices and virtual applications.

The AIP uses **mTLS** (mutual TLS), an extension of TLS, which is basically a TLS with a two-way verification. When we say “client”, in the context of the AIP, it means an HTTP, MQTT, etc. request made on the behalf of the user and thing - a person or an organisation (user) requesting a read/write access to a certain device or application (thing). When we say “server”, in the context of the AIP, it means the Mainflux messaging middleware itself, i.e. a software that handles and responds to client requests.

The identities of both server and client are verified before a connection is established and the encrypted data is exchanged in both ways. This means that not only the server, as described above, but also the client must have and submit a certificate that the server should recognize before proceeding on to the HTTP or MQTT message exchange. The implications are two-fold.

Firstly, a secure place to store certificates and an automated way of certificate life cycle management are required. The AIP currently exposes a command-line interface (CLI) to issue certificates. However, the AIP supports the use of [Vault](#), an open source and identity-based secret and encryption management system used to safely and securely store and protect sensitive data contained in certificates.

Vault (Figure 2) uses a standardised procedure for distribution and lifecycle management of cryptographic keys, that is, users' and things' SSL certificates.

Secondly, an automated way to issue certificates is required. Vault's Public key infrastructure (PKI) secrets engine dynamically generates X.509 certificates (used by Mainflux) on demand. This allows users and things to acquire certificates without going through the usual manual process of generating a private key and Certificate Signing Request (CSR) via CLI.

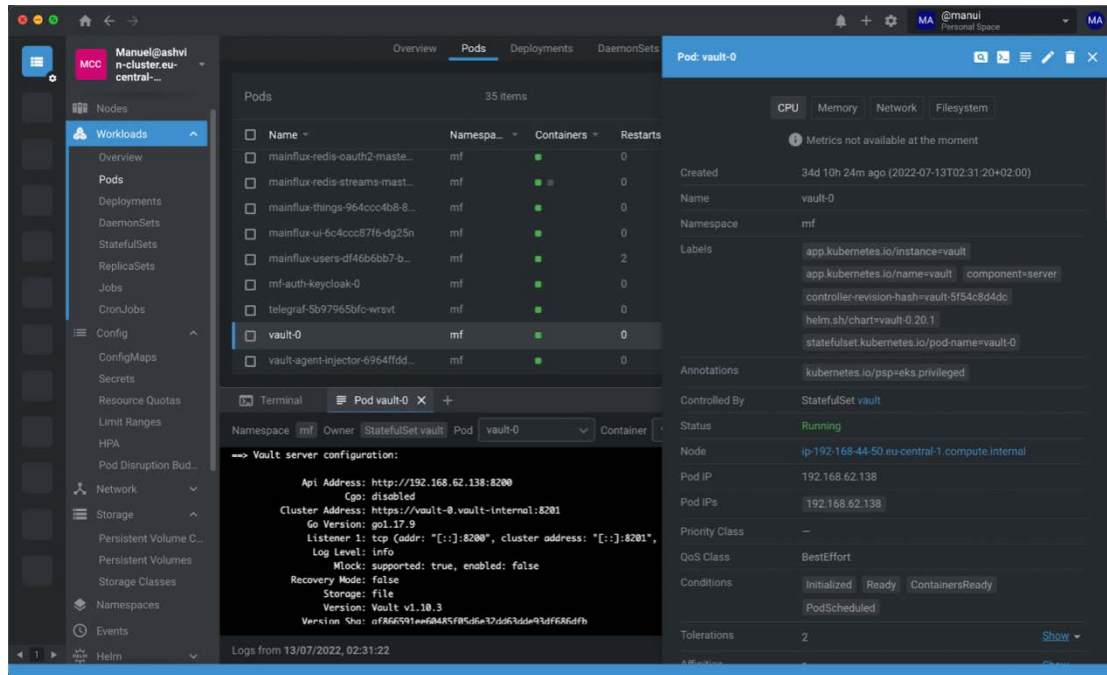


Figure 2: Kubernetes cluster with Vault hosted in Kubernetes pod

SSL certificates are used to secure a TLS server-client connection. Simply put, an SSL certificate is a file stored in a special purpose database on the server side. An SSL certificate contains a thing's **public key** accessible to the AIP platform users. Besides the thing's public key, a thing's **private key** is kept secret and secure. The thing's private key is accessible only by Mainflux middleware, i.e. on the server side. The user uses the public key to encrypt the message read by the server (more precisely, the things service, one of the Mainflux middleware microservices).

By successfully decrypting a message that was encrypted with the public key - please notice the message encrypted with the public key can be decrypted only with the private key - the server proves that it possesses the private key and thus proves its **identity** to the user.

3.3.2 Single sign-on and user management with Keycloak

Our IoT Platform based on Mainflux middleware is a game engine-based digital twin platform supporting various design and engineering applications. Mainflux middleware is a set of microservices, which includes auth service.

Auth service is responsible for both authentication and as well as for authorization. Most of these components including IoT Platform and digital twin platform, require user **authentication**. Each component has its own separate user management system and a separate authentication mechanism. This creates a problem of organising and maintaining separate user identities and credentials. To solve this problem, a single sign-on (SSO) system is used. SSO is an authentication mechanism that allows a user to log in with single credentials to any of the aforementioned parts of the ASHVIN platform.

Behind the authentication wall, a user should be able to access freely everything it is authorised to access. The access clearance mechanisms belong to the topic of authorization which is not to be confused with authentication. The latter simply determines whether someone or something is who or what it says it is, the former is about read, write and execute permissions over digital entities such as files.

To implement a **single sign-on (SSO)** mechanism, our platform uses [Keycloak](#), an open source identity and access management system (Figure 3).

Keycloak provides:

- user federation - the way to associate a person's or organisation's identity and attributes across multiple different identity and attribute management systems; in the context of the AP, it means to associate single user's identities of the AIP, DT and accompanying tools,
- strong authentication - multi-factor authentication (by using, e.g., an email/phone message and a password-username credentials) and challenge-response protocol (where a user must provide a valid answer to a question, e.g. to recognize a number on a picture),
- user management - the complete user CRUD (create, read, update and delete) system
- fine-grained authorization, e.g. time-based access control, and more.

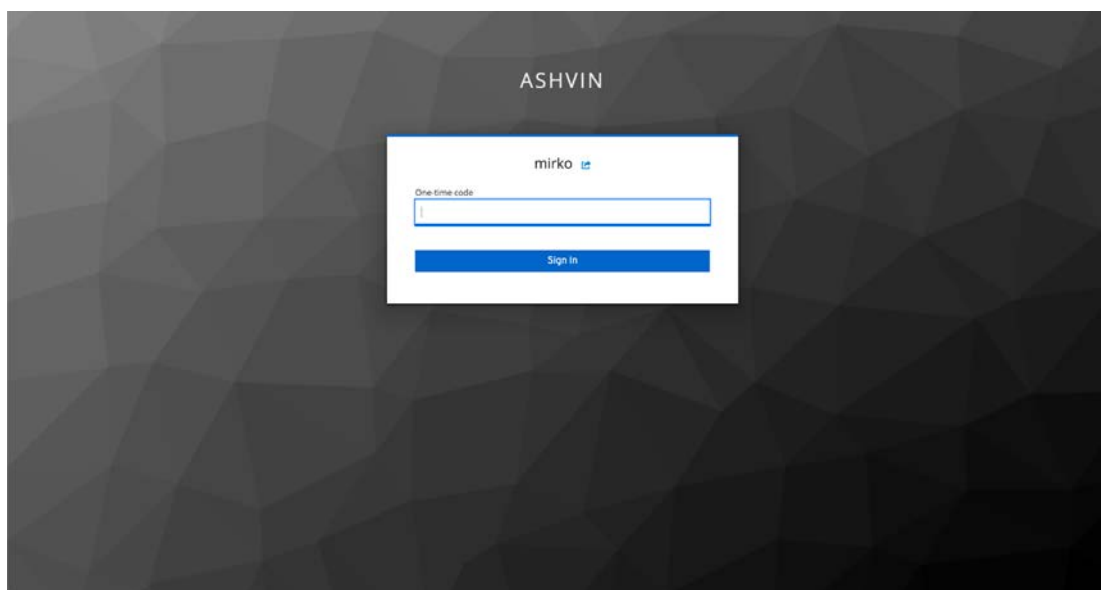


Figure 3 Keycloak login screen

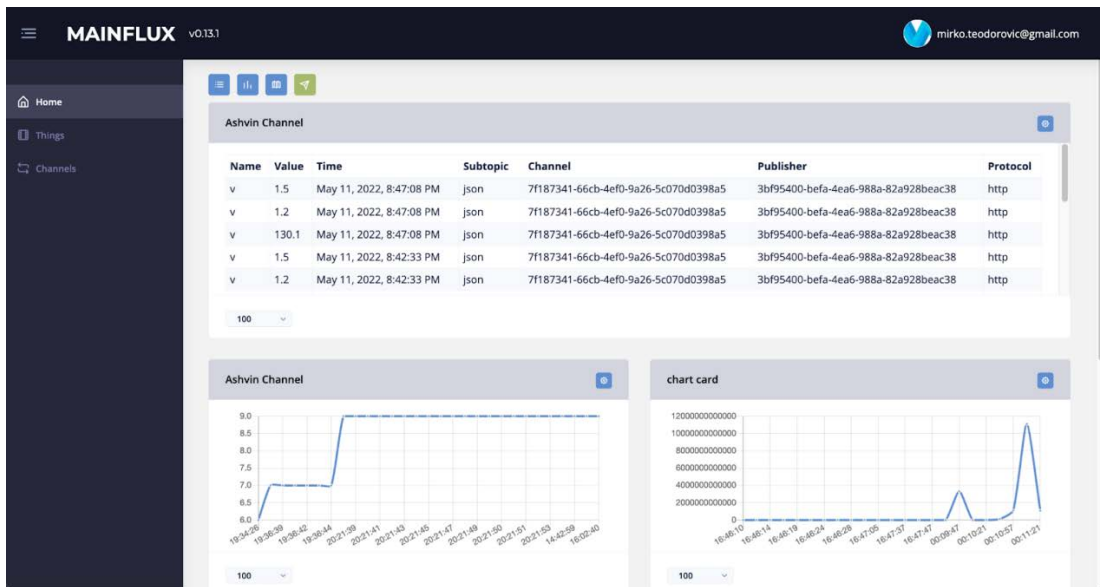


Figure 4 Mianflux Dashboard screen

In concrete terms, this means that the AP users don't have to deal with login forms, user authentication and other complicated aspects of user identity and credentials management per the AP component.

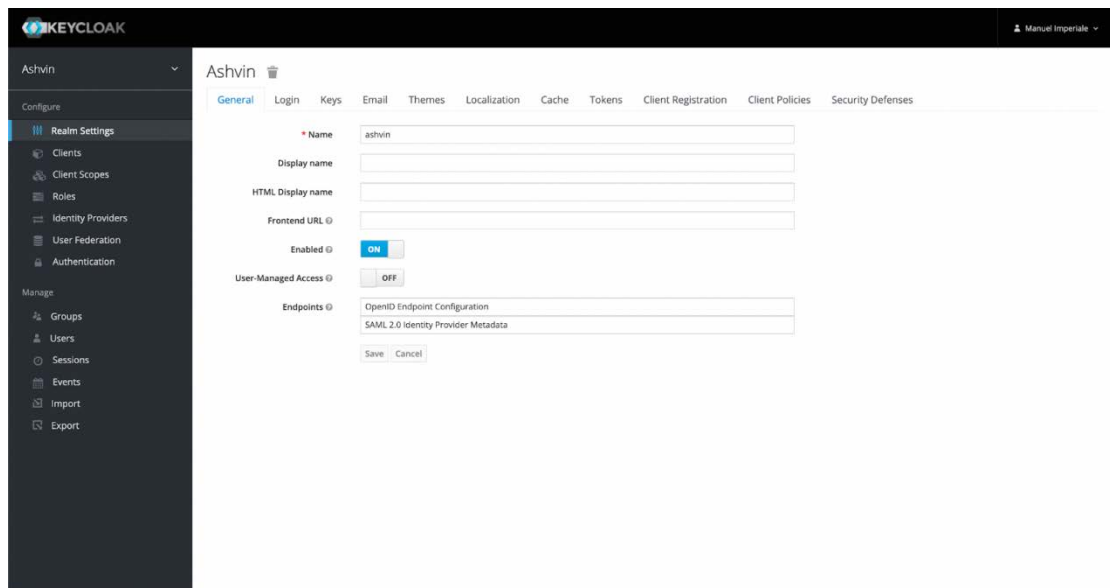


Figure 5 Keycloak admin panel

The opposite of the sign-in is the sign-out. The problem with the non-single sign-out is that a user can log out of one application, say, the DT, while staying logged in another application, say the AIP, forgetting to log out, thereby creating a security breach. When there are many more applications, as in our case, potential for security compromise increases proportionally. To solve this problem, Keycloak features a single sign-out mechanism. Once logged out of Keycloak, users cannot access anymore anything which resides behind the Keycloak virtual protection wall.

3.3.3 Open weather data gathering, processing and aggregating with Telegraf

Telegraf is a server agent for gathering, processing, aggregating, and persisting metrics (measurements). It is used as a plugin which integrates into the existing system, the AIP, in our case, and allows developers to add support for metric collection. The metric collection mechanisms added by Telegraf complement the already existing metric collection mechanisms used by a system (the AIP). As suggested, Telegraf not only enables you to gather data, it also transforms, decorates, and filters metrics. Finally, with Telegraf, metrics can be aggregated thereby applying basic descriptive statistics (e.g. mean, min, max, quantiles, etc.) to the collected data. Together with Grafana, an open source analytics and interactive visualisation framework, Telegraf presents a powerful option for data description and visualisation.

The AIP uses Telegraf in order to import [OpenWeatherMap](#) data, an online service that provides forecasts, hyperlocal precipitation forecast, nowcasts and historical weather data for any geographical location via API (One Call API 3.0). Figure 5 shows the three types of data collected (historical and current) and generated (forecasted) by the OpenWeatherMap.

Telegraf uses OpenWeatherMap's One Call API 3.0 in order to get weather data for a specified location. For now, the data is gathered for an international Airport at Zadar, Croatia, but there is a plan to continuously gather data for all geographical locations of demo sites.

Telegraf is compatible with InfluxDB, an open-source time series database, used by the AIP to store time series data, such as weather data. So, once Telegraf picks up the data, it is forwarded to the InfluxDB database residing in the AIP Kubernetes cluster. Subsequently, weather data is available as a real-time stream or as historically stored time series (Figure 6).

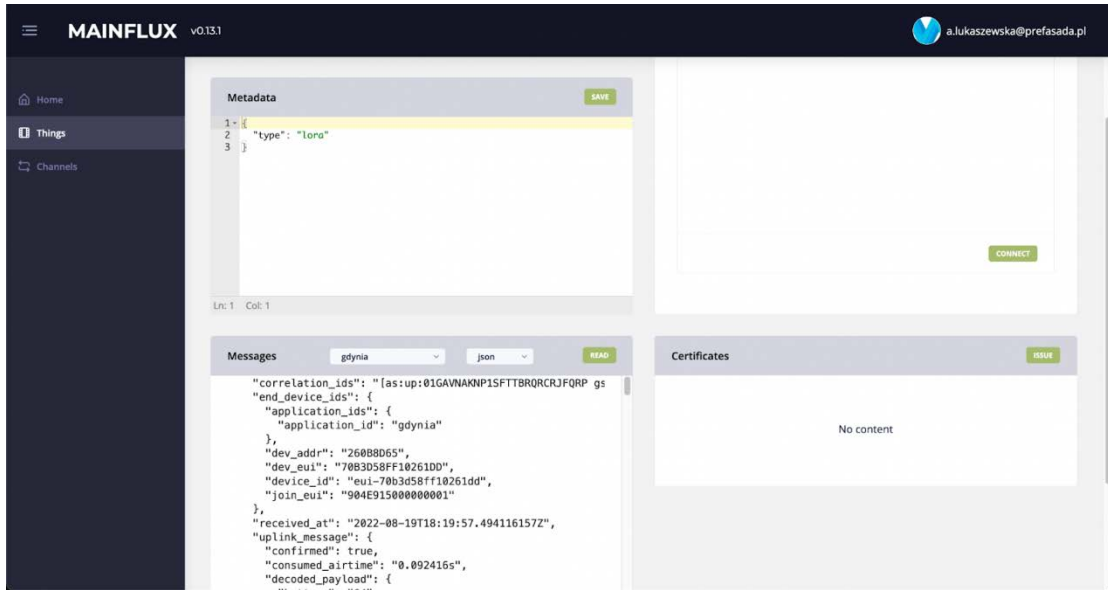


Figure 2 Weather data in JSON in Mainflux UI

3.3.4 User and thing groups authorization policy management with Ory Keto

The AIP is built around three main entities: users, things and channels. Users represent real (human) users or organisations. Things represent physical devices (on the edge) or virtual applications (in the cloud or on-premises) connected to Mainflux. Connected applications and devices use the AIP for message exchange with other "things" (apps and devices) via channels. Channels refer to communication channels between device/application abstractions, i.e. things. Mainflux channel is basically a message topic (which translates to an MQTT topic, HTTP address) that can be used as a message relay by all the things connected to it.

For grouping the AIP entities, there are groups object in the auth service of Mainflux middleware (the AIP backend/server service). The auth service is responsible for both authentication and authorization. Groups are primarily used to group users (persons and organisations) and things (devices and applications) according to arbitrary criteria to achieve logical organisation. Groups are organised like a tree, akin to a family tree. Group can have one parent and possibly zero or more children. There is no upper limit to the number of children a group can have. A Group with no parent is considered the root of the tree structure. Whereas Groups are yet implemented in the user interface (UI) mode, they are fully functional in the backend of the AIP. Further steps will implement group management through UI. Figure 7 - Figure 9 show our UI Groups.

In the AIP, we set **permissions** on the AIP entities: users, things, and groups of users and things. User's permissions control user's ability to create, read, update and delete things and other users. Thing's permissions control things read/write access to channels.

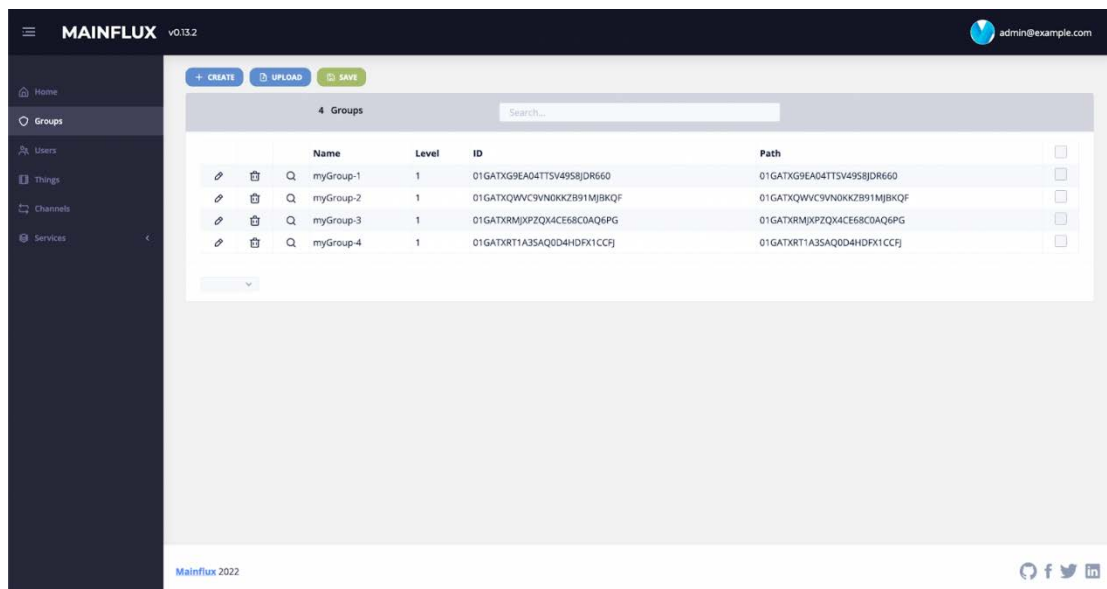


Figure 3 Mainflux UI Groups

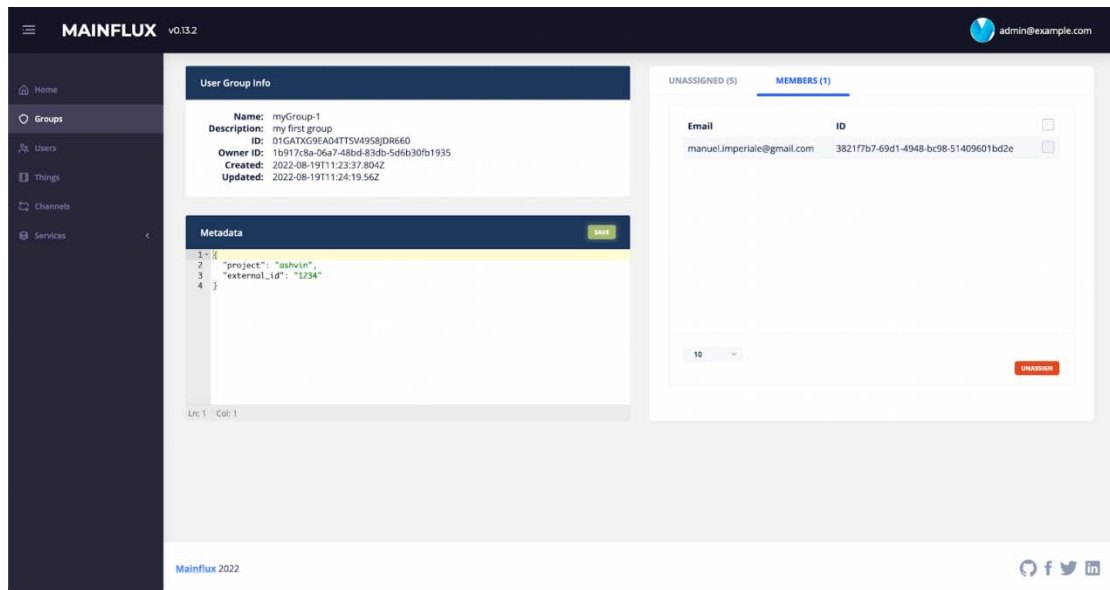


Figure 8 Mainflux UI Groups

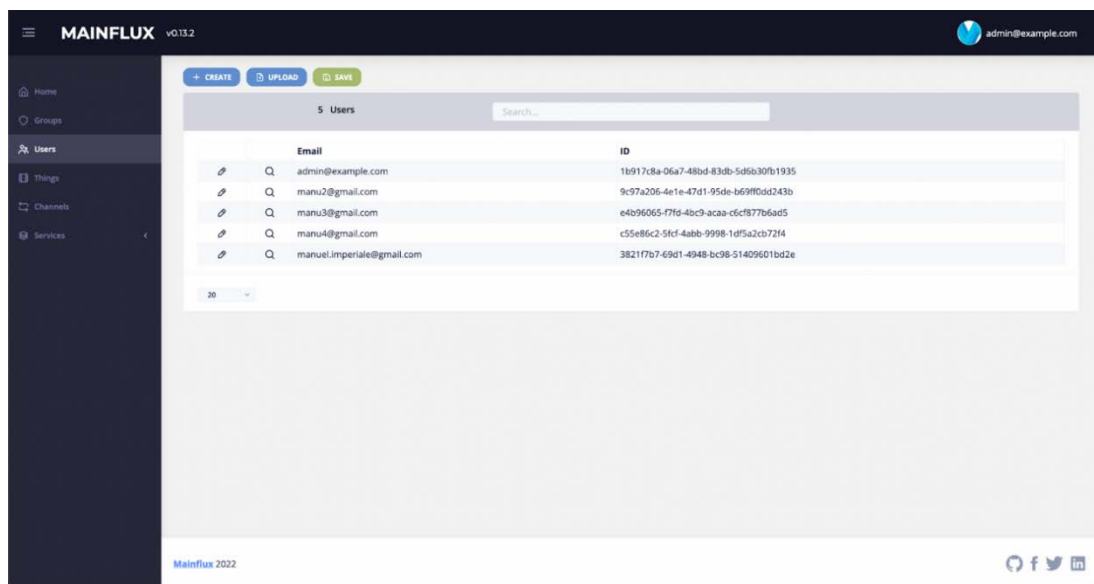


Figure 9 Mainflux UI users and users details

The AIP policies are based on [Ory Keto](#), an open source Golang implementation of [Zanzibar](#), Google's global authorization system. The policies used by the AIP constitute what is known as **relationship-based access control (ReBAC)**, where permissions are organised based on relationships between digital resources, i.e. users, things, and groups of users and things:

- Subjects that bear the policy such as users or things, or groups of users and things.
- Objects the policy bears upon; again, users or things, or groups of users and things.
- Relation, an action that the subject wants to do on the object, such as CRUD (create, read, update and delete) or read/write.

The subject-object-relation set constitutes the anatomy of a single policy. On the other hand, The concrete AIP set of policies include:

- **member**: used to indicate memberships of users, things and groups (of users and things), e.g. if this user belongs to that group,
- **read, write**: the permissions users and groups of users have in relation to things and groups of things, e.g. if this user is allowed to publish messages (write) *via* on behalf of that thing,
- **create**: allows a creation of new users, i.e. if this user is allowed to create a new user.

3.3.5 DigitalOcean to AWS Kubernetes migration and automated DB management with AWS and Velero

The AIP uses DigitalOcean cloud computing services to provision and deploy its services containerized by Docker and clusterized via Kubernetes. Docker technology is used to put services in virtual OS based packages called containers. Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications in the form of configurable clusters of nodes. Therefore, the user accesses a node of the Kubernetes cluster which consists of a set of pods where containers with a desired service (such as database retrieval of messages) is located.

Migration took place so the AIP Kubernetes cluster (Figure 9 and 10 and 11) is hosted on Amazon™'s AWS cloud computing platform which offers a set of infrastructure as a service (IaaS), platform as a service (PaaS) and packaged software as a service (SaaS) solution. In addition, it can handle extensive traffic which is well suited for our needs of IoT messaging middleware. (e.g. when sending/receiving images).

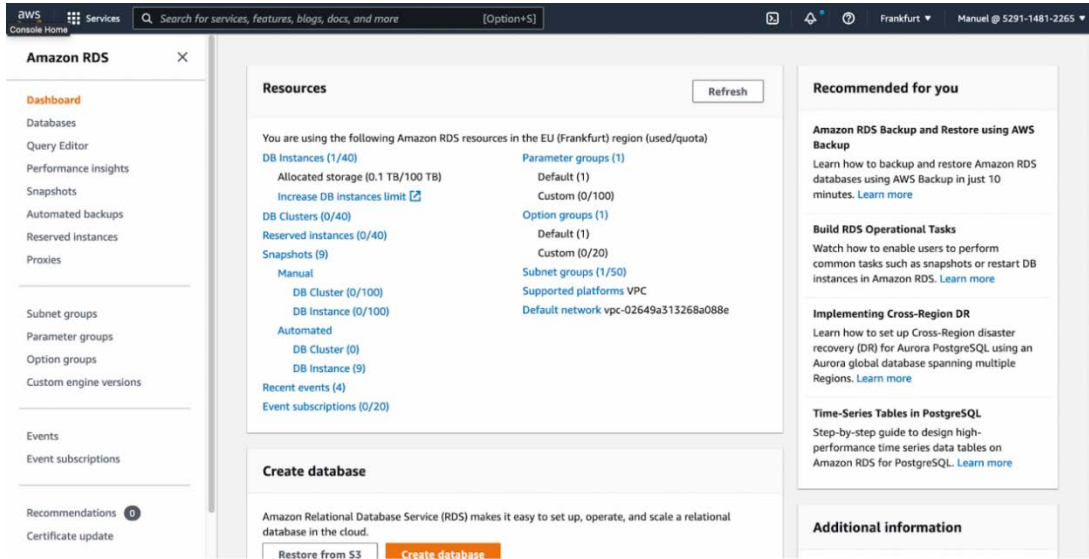


Figure 10 AWS Kubernetes cluster admin panel

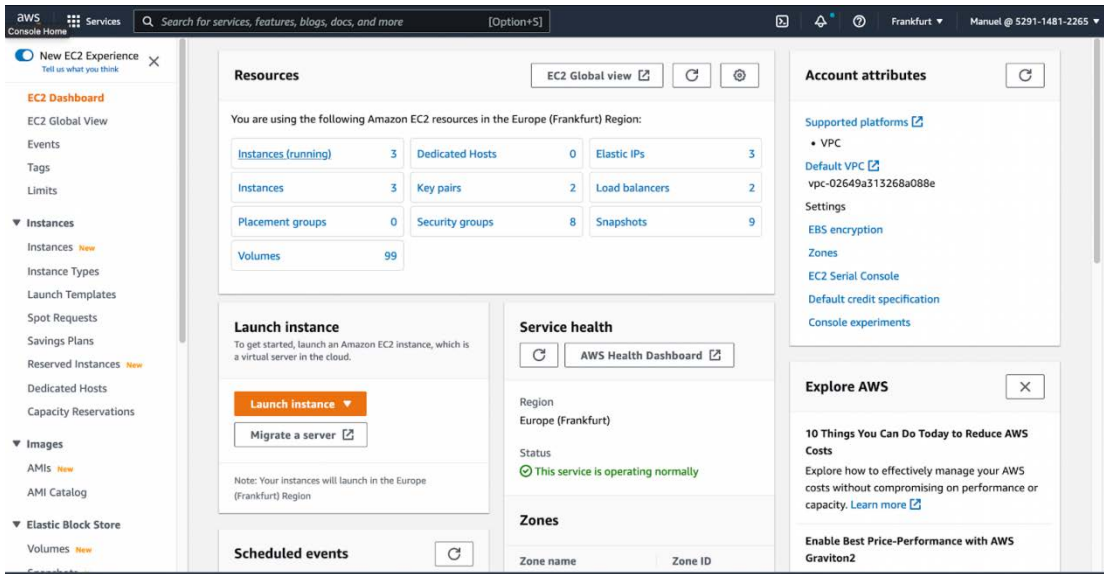


Figure 11 AWS Kubernetes cluster admin

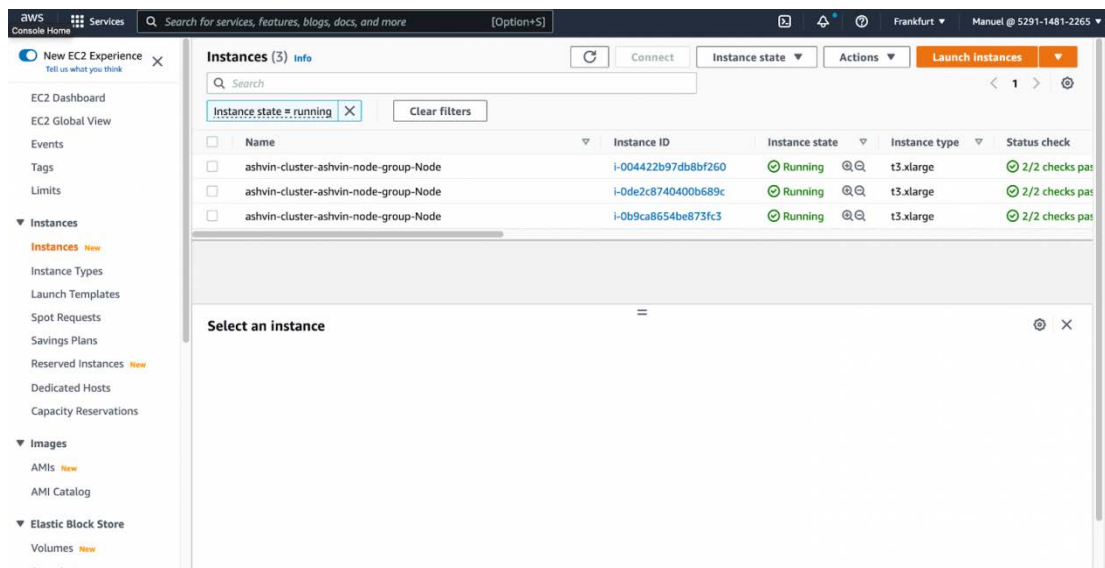


Figure 4 AWS Kubernetes cluster admin panel

AWS also offers fully managed database services. This provides for continuous monitoring, self-repairing, and automated scaling of database related services as well as databases themselves. The AIP is based on the Mainflux IoT messaging middleware and is thus built around entities such as users, things, groups of users and things, and channels. Besides these entities, there are also messages that are exchanged by things via channels. Users, things, groups of users and things, channels and messages need to be stored in respective and different databases. That means that the AIP needs, in order to function properly, at least half a dozen database volumes (see below) and related database microservices (see below).

AWS managed database system does the handling of the database management and storage in a fully automated manner. Instead of users manually handling services and volumes, the AWS managed database system does the handling "behind the scenes" and simply exposes a single URL (an address to access a database) to the rest of the AIP Kubernetes cluster.

On top of that, the AWS managed database system enables Relational Database Service (RDS) managed database encryption (Figure 12). An encrypted DB instance provides an additional layer of data protection by securing data from unauthorised access to the underlying storage. The encrypted data keeps, behind the "wall" of encryption, the underlying storage (basically, previously mentioned volumes), thus enabling automated backups, read replicas, and snapshots of databases.

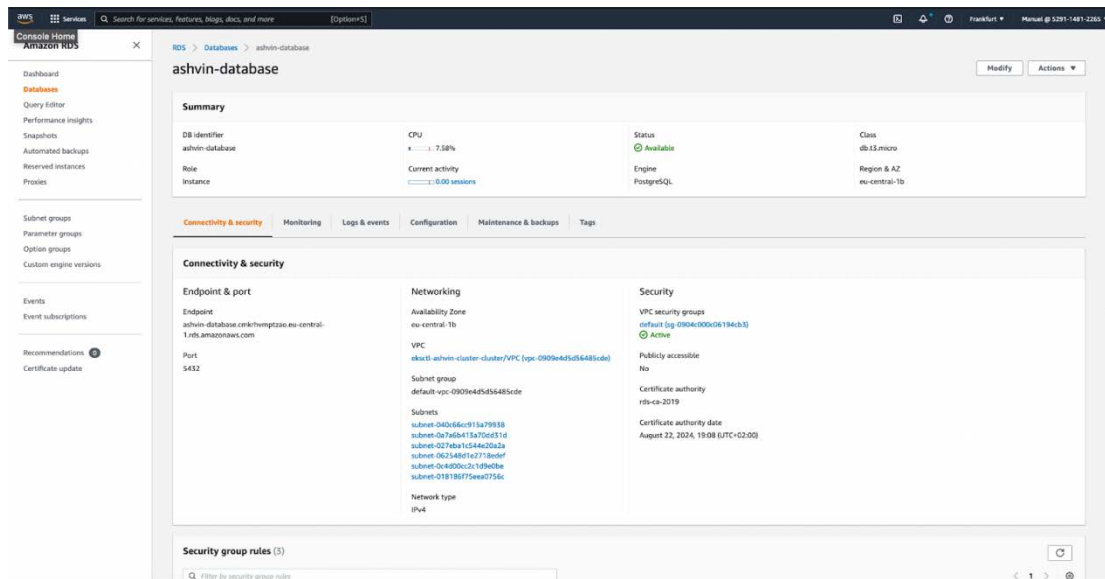


Figure 5 AWS managed DB admin panel

The downside of the AWS managed database system is that it does not support InfluxDB, a time series database, used, amongst other types of databases, by the AIP to store primarily messages. However, the AWS managed database system does not handle time series databases.

As a consequence, we still need to manage InfluxDB in a semi-manual fashion described, i.e. by using a specialised type of service (a database manager) and a connected volume (a database physical storage). However, the process of backup and restore, disaster recovery, and migration of InfluxDB based databases is done using an open source tool [Velero](#). Velero is specifically meant for Kubernetes cluster resources and persistent volumes management automatization. To put it simply, Velero "wraps" around InfluxDB related services and connected volumes and offers a similar set of operations and automations to AWS managed database system.

3.3.6 Indoor environmental data gathering, and aggregation with ChirpStack LoRaWAN Network

The LoRaWAN protocol is a Low Power Wide Area Networking (LPWAN) communication protocol based on LoRa, a wireless modulation technique based on Chirp Spread Spectrum (CSS) technology. CSS is a wireless audio frequency technology that operates in a licence-free radio frequency spectrum. It is robust against noise and disturbances, and signals can be sent and received (relayed) across long distances. The LoRaWAN itself is an open specification, so anyone can set up and operate a LoRa server network.

The AIP supports use of LoRaWAN Networks by means of lora-adapter microservice. The lora-adapter service is located between the Mainflux middleware (the AIP backend) and a LoRa Server (located in LoRaWAN network). The adapter forwards messages from a LoRa Server to the Mainflux channels via MQTT protocol, using the adequate MQTT topics and the appropriate message format (JSON and SenML), i.e.

respecting the APIs of both systems (LoRa Server, on the one hand and the Mainflux middleware, on the other).

In the context of the AP, LoRaWAN Network and LoRa Servers are used to collect data from indoor environment by means of a set of sensors. Sensors are connected to the router which routes the data to the LoRa Gateway. The latter relays the data to the LoRa Server found in the cloud. The whole LoRa "on the edge" setup (see Figure 11) is located in a public two-story residential building in Gdynia, Poland. The building was constructed in 1921 and has a very low energy performance. Indoor environmental data can provide further insights related to heat loss and thus can be used to improve the energy consumption balance of the building.

The first attempt to collect data and store it in the AIP was done using [The Things Network](#), a LoRaWAN Network Server provider. The Things Network is based on [The Things Stack](#), a complete and open source LoRaWAN Network Server technology. The Things network exposes an interface to securely manage applications, devices and gateways. It also offers a set of tools to manage and use LoRa servers. The indoor environmental data federated by means of LoRa Servers hosted by The Things Stack was picked up by the aforementioned Mainflux middleware lora-adapter and forwarded to the InfluxDB database located in the AIP Kubernetes cluster.

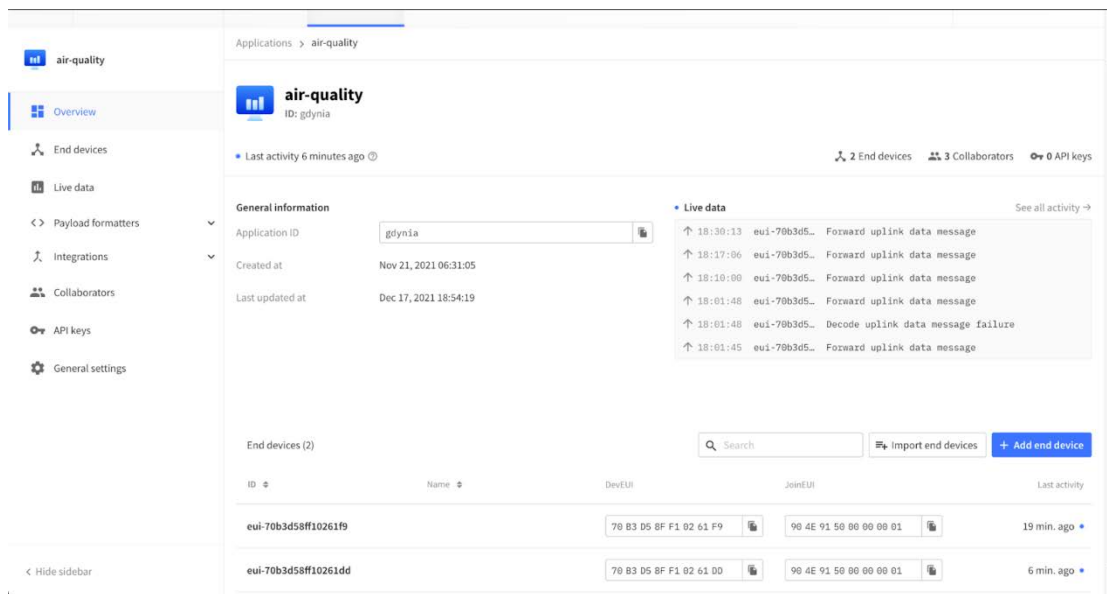


Figure 6 The Things Network LoRaWAN Network Server admin panel

ASHVIN's in-house customised LoRa Server was deployed using the [ChirpStack](#) open-source LoRaWAN Network server stack (Figure 13 and Figure 14). The indoor environmental data is now directed to LoRa Servers found on <https://lora.mf.ASHVIN.eu/> address. As in the case of The Things Network, the data on the LoRa server is picked up by a lora-adapter and forwarded to the InfluxDB database found in the AIP Kubernetes cluster.

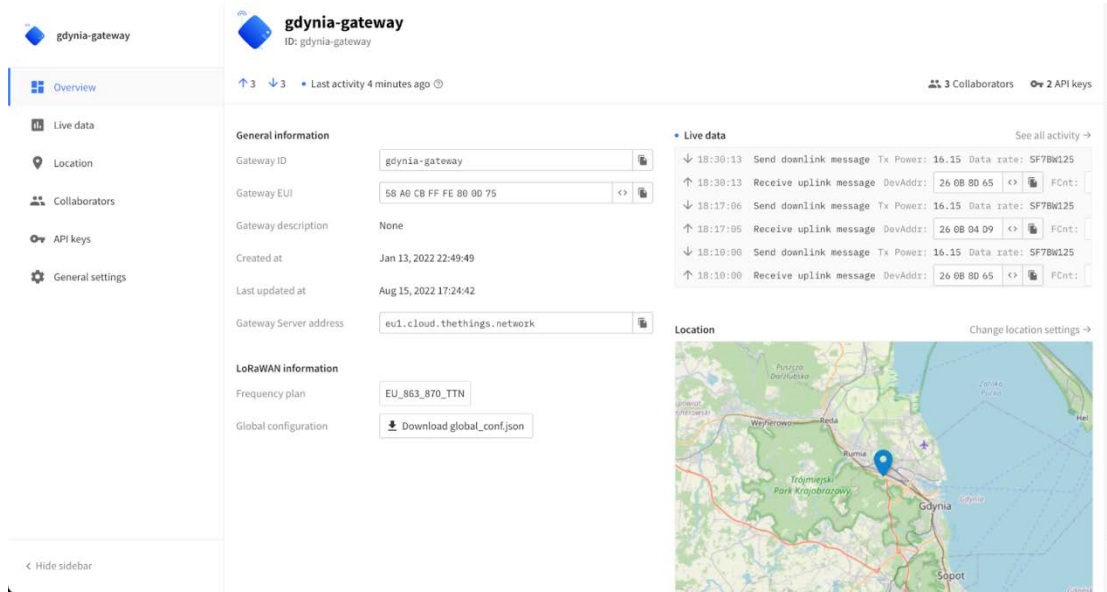


Figure 7 The Things Network LoRaWAN Network Server admin panel

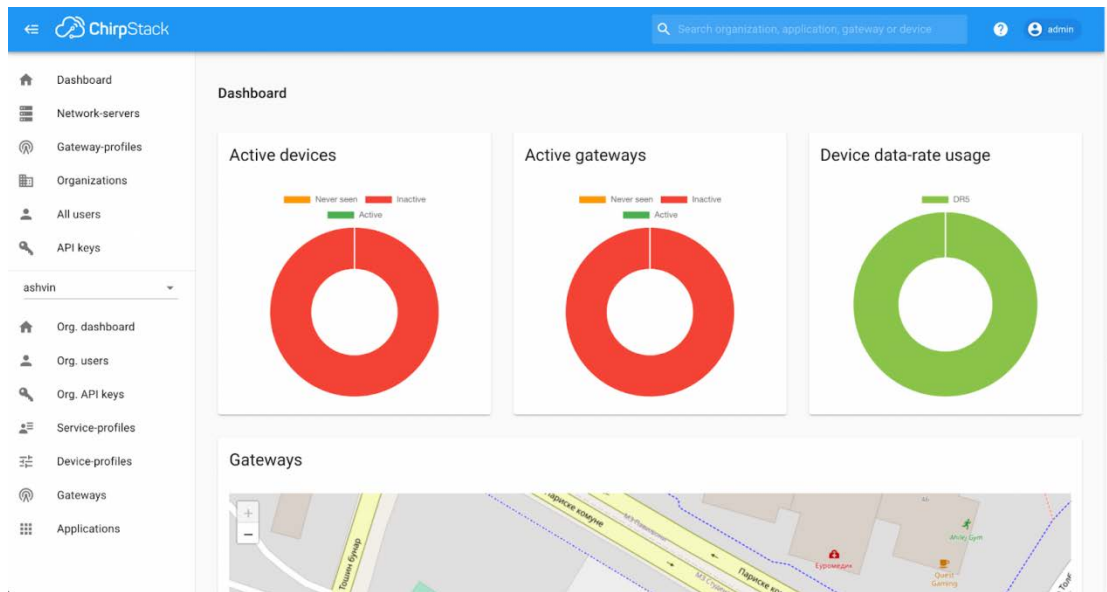


Figure 8 ChirpStack admin panel

4. DATA FUSION METHODS

Task T3.4 of data fusion is responsible for the analysis and fusion of heterogeneous data collected from pilot sites. Data fusion will be implemented in some of the pilot use cases, serving different purposes each time. Data fusion methods support Digital twin (DT) based IoT data processing pipelines by providing the result of the prediction algorithms, e.g. predicted number of people in a pilot site.

4.1 Goals and objectives

The main outcome of the task is the development of the necessary algorithms for collecting, processing and fusing data/information from various sources, which are mainly sensors. The data/information from various sources across the different pilot sites is heterogeneous therefore, the main is its combination (fusion) using suitable methods.

Fusion is a broad term that includes methodologies that combine heterogeneous data/information either at a results level (late fusion) or at data level (early fusion). Unlike late fusion which has a wider variety of techniques, early fusion is predominantly achieved by concatenation. Concatenation uses homogenous formats of pre-processed and filtered data/features extracted from the raw variables, as input in a predictive algorithm. Late fusion refers to the combination of predictive results from different data sources. For example, let's say that there are two sensors installed in a construction site for fall detection and predictive algorithms are applied on each of these sensors, then the prediction results of the two algorithms are combined with a proper late fusion method, in order to extract a firmer result about fall detection.

Our pilot sites from which the data is sourced are summarized hereunder.

- **PUC2 (FASADA):** This PUC2 in Poland uses the data/information from environmental sensors installed in a residential building for occupancy detection and to assess the air quality. The ground truth for this prediction will be an integral variable revealing the number of people per hour on certain days. This information is received by questioning the habitants of the building.
- **PUC5 (NCC):** This pilot use case is the construction of an office complex in Sweden which collects the data from installed environmental sensors for **occupancy detection** to understand the level of activity per room or per floor. The installed sensors also provide information about the activity level, with a continuous variable that is used as a ground truth for the development of algorithms.
- **PUC6 (UPC):** This PUC assesses the crane's positions and activities during construction using TUB-developed classification algorithms, based on data from a device mounted on the crane in Spain. The crane-mounted-device encapsulates GPS, accelerometer, gyroscope and pressure sensors.
- **PUC10 (NGEO):** This PUC is a quay wall in the Netherlands. The goal here is to combine the data/information from the available sensors in order to find the correlation patterns between water levels inside and outside the quay wall.

Section 4.2 presents the applications of data fusion by analysis categories. The first category of activity recognition and occupancy detection problems which involves

solving classification or regression algorithms is followed by the second category of implementing the correlation patterns' analysis and crane activity tracking.

4.2 Activity recognition and occupancy detection

This section describes the work that has been done until the preparation of the current deliverable. The following subsections are divided according to the type of the task performed on PUC2 and PUC5.

4.2.1 Occupancy detection through environmental sensors

i. Related work

In smart buildings, indoor air quality systems can be used not only to estimate the quality of the air condition, but also to estimate the mobility and presence of people/objects, a task known as occupancy detection. In a nutshell, occupancy detection involves detecting the presence of people in a space (building or room), which can be useful for safety reasons, such as intrusion detection or presence of people in a construction site where dangerous operations are taking place.

Indoor air quality systems consist of environmental sensors that are non-intrusive and easy to install and use. Such sensors measure the humidity level, the presence of smoke, the dust concentration in the atmosphere, and temperature among others.

Machine learning and deep learning algorithms are data driven approaches used to predict occupancy from environmental sensors. Support Vector Machines or KNN algorithms applied to sensor data have several applications in the relevant literature. There are a variety of sensor combinations described in the literature, but there is no apparent pattern indicating that adding more sensors to an algorithm improves its accuracy. These applications may also incorporate methods for feature selection and extraction. Different combinations of sensors can be found in literature, without having a clear pattern of improving the accuracy results when using more sensors in the algorithm. These applications may also include feature selection and extraction methods.

ii. Implementation

PUC2 and PUC5 are explored for occupancy detection using environmental sensor data collected from a real physical environment and not a simulated experiment. However, it is expected that real environment applications lack adequate ground truth values. In PUC5 the ground truth was measured from a device as a quantitative variable that denoted activity. For PUC2, the ground truth was collected through questionnaires completed by the habitants of the building who stated their presence for specific days and time slots.

The specific environmental sensors in PUC2 comprised of Humidity, CO₂, Light, Pressure, Temperature, VOC (gas detector) whereas PUC5 sensors comprised of Smoke, Sound Level, Temperature, Relative Humidity, Dust, Air Pressure, Activity, Acceleration.

For both use cases, the implementations include concatenation of the sensor values and application of predictive algorithms.

For PUC2 (FASADA), the following is a summary of the actions that were taken to attain the analysis' results:

1. Collect sensor data from online database.
2. Receive target variable through a questionnaire.
3. Transform and merge online and questionnaire data.
4. Clean and pre-process.
5. Create and test a variety of models.

In PUC2, data was transformed into a format more conducive to the application of machine learning models and the prediction of the building's occupancy. The target variable (i.e. the variable to be predicted), which was obtained through questionnaires the occupants were required to answer, was converted from its original format into a data frame which was then fused with the sensor data to deliver the essential data to the models. After the data transformation was complete, data and pattern analysis was conducted to identify potential connections between the variables. Figure 15 displays part of the data frame that was created after the data transformation and was used for the machine learning model creation.

battery	co2	humidity	lux	pressure	temp	voc	value
85.0	569.0	38.5	14.0	1017.01	22.63	149.0	0.0
85.0	585.0	37.5	14.0	1017.01	22.34	212.0	0.0
85.0	816.0	40.0	14.0	1016.09	22.87	123.0	0.0
85.0	569.0	38.5	14.0	1017.01	22.63	149.0	0.0
85.0	585.0	37.5	14.0	1017.01	22.34	212.0	0.0
85.0	816.0	40.0	14.0	1016.09	22.87	123.0	0.0
85.0	569.0	38.5	14.0	1017.01	22.63	149.0	0.0
85.0	585.0	37.5	14.0	1017.01	22.34	212.0	0.0
85.0	816.0	40.0	14.0	1016.09	22.87	123.0	0.0
85.0	569.0	38.5	14.0	1017.01	22.63	149.0	0.0
85.0	585.0	37.5	14.0	1017.01	22.34	212.0	0.0
85.0	816.0	40.0	14.0	1016.09	22.87	123.0	0.0

Figure 9 Final Data format (PUC 2)

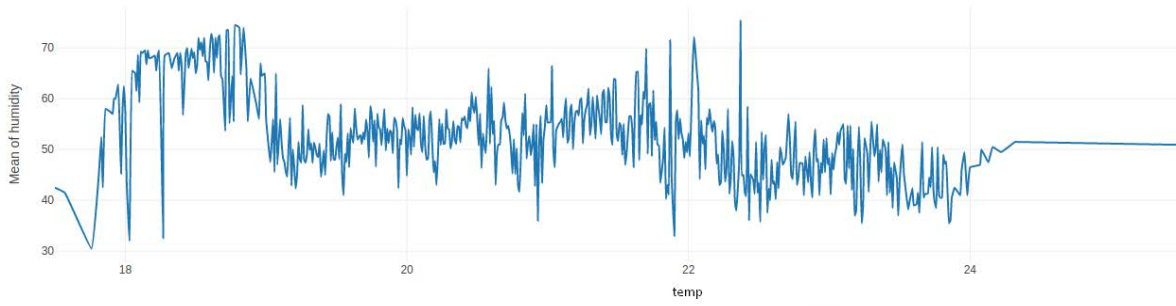


Figure 18 Fluctuation of mean humidity value with the change of temperature (PUC 2)

Figure 18 and Figure 19 give a concise overview of how the data was utilised prior to model training. The correlation between the characteristics was investigated, as was the significance of the characteristics. Figure 17 illustrates the significance of each of the six utilised attributes. The most crucial features are 2, 3, and 5, which relate to humidity, lux, and temperature.

	battery	co2	humidity	lux	pressure	temp	value	voc
battery	N/A	-0.14	0.17	0.25	-0.05	0.64	-0.26	0.21
co2	-0.14	N/A	0.32	-0.41	0.00	0.35	-0.19	-0.12
humidity	0.17	0.32	N/A	-0.42	-0.48	0.29	-0.28	-0.11
lux	0.25	-0.41	-0.42	N/A	0.03	-0.06	0.23	-0.03
pressure	-0.05	0.00	-0.48	0.03	N/A	0.36	-0.20	-0.23
temp	0.64	0.35	0.29	-0.06	0.36	N/A	-0.49	-0.10
voc	0.21	-0.12	-0.11	-0.03	-0.23	-0.10	0.00	N/A
value	-0.26	-0.19	-0.28	0.23	-0.20	-0.49	N/A	0.00

Figure 19 Correlation of sensor values (PUC 2)

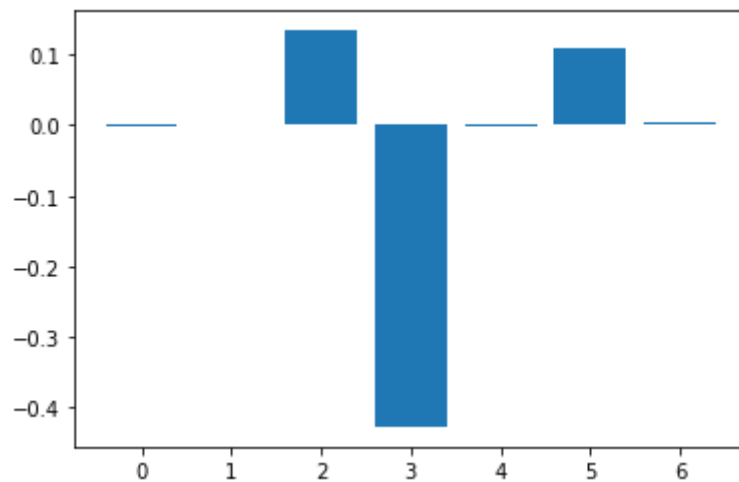


Figure 20 Feature importance (PUC2)

For the FASADA use case a variety of regression models were used. The best performance derived from CatBoostRegressor with $MSE = 0.06$ and $R^2 = 0.63$, the second best model was Extra Trees Classifier with similar metrics. In this case, the available data for the model are limited, and as a result, the preliminary findings cannot be relied upon.

Following with the analysis conducted on PUC5 (NCC), the following actions were taken:

1. Downloaded data from online storage.
 2. Analysis of files from various dates and selection of the best one.
 3. Pre-process accordingly the missing values.
- Create and test a variety of models.

Since the recording of data is done on a monthly basis for this PUC, there is plenty of data for model building and causality analysis, with the downside of a lot of missing values. Figure 21 depicts the case with the fewest missing values. From Figure 18 a problem that often occurred is evident, were in most of the cases there were a lot of missing values either on the sensors or the target variable. This use case has a number of distinct files from different dates that were analysed in order to identify the file containing the most relevant data for our experiment.

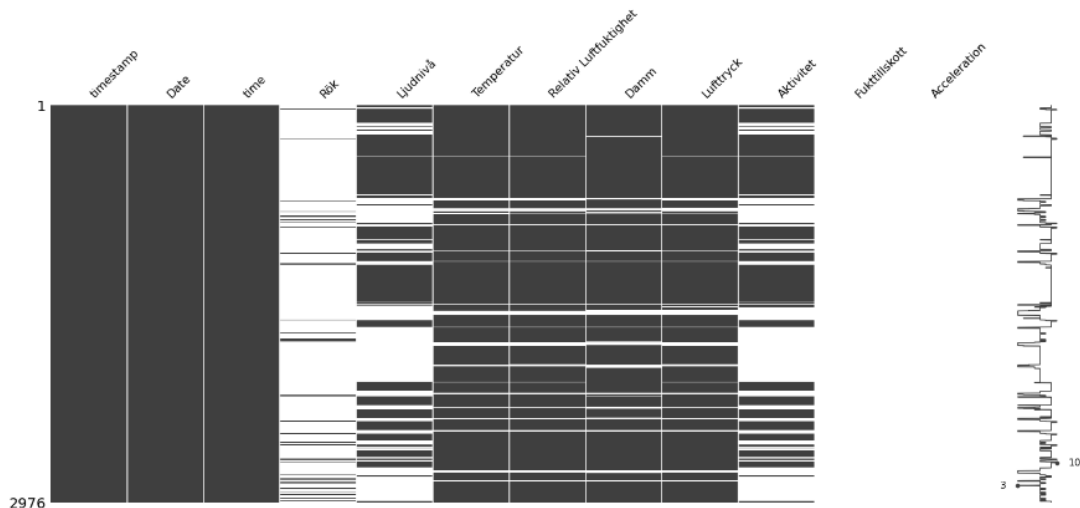


Figure 21 Missing/NaN values (PUC 5)

Due to the nature of the data and the number of missing values, the MAE and MSE values were too high for each of the models that were utilised to identify the model with the most accurate predictions.

In this instance, K Neighbours Regressor and Extra Trees Regressor performed the best. In both instances, the MAE is near to 2 and the MSE is close to 20, making the models less dependable than desired. Simpler and faster models, such as Linear Regression failed as the MSE was ten times larger. Due to poor performance of the models in this use case, no results are offered in this deliverable.

4.2.2 Activity recognition for duration extraction using cranes

On the demonstration site #6 in Barcelona, kinematic sensor data were collected during concrete pouring works executed by a tower crane. Different sensors were mounted on a crane hook and the collected data were fused for data mining. The raw data were classified by ML classifiers for activity recognition to extract activity durations. Due to continuous data collection, it could be investigated how the activity durations change according to an increasing amount of gathered data.

Method

The proposed method is presented in Figure 21. At first, raw data are collected during construction execution by various kinematic sensors. The collected raw data are sent via the internet automatically to the ASHVIN platform. On the ASHVIN platform, it is possible to access the collected raw data and fuse them. The data set has to be checked for duplicate timestamps and missing data points have to be interpolated to create a continuous data set with the same distance between the data points. The data are normalised and several data points are merged to sliding windows. The sliding windows are used to calculate features for a number of data points. Afterwards, the sliding windows have to be labelled by comparing the timestamps with a recorded video. Then different ML classifiers can be used for supervised learning. Subsequently, these ML algorithms are used for classification of the data into different labels – the different activities. The performance of the classifiers has to be compared as there is no general best ML algorithm. The results of the best performing classifier can be used for activity recognition. Each classified instance is equal to one window. The duration of each repetition of each operation can be calculated by: $n \cdot w_l \cdot o_l = \text{duration}$, with n = number of windows, w_l = window length, and o_l = percentage of overlap. Afterwards, heuristics are used to detect non-meaningful classified instances. This results in the final operation durations. If the construction execution starts again, new data are collected and the whole procedure begins again.

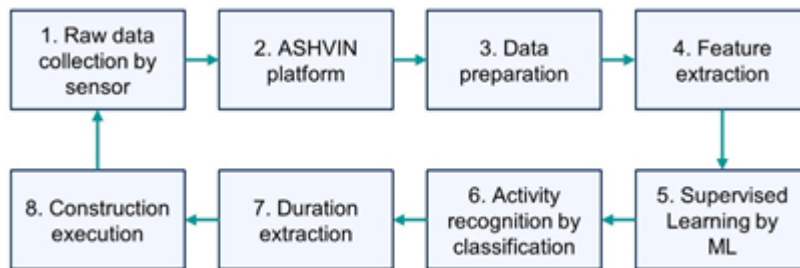


Figure 10 Methodology pipeline

Data collection

The WTGAHRS2 device consisting of an inertial measurement unit (IMU), a GPS tracker, and a barometer was mounted on a crane hook during concrete works for data collection (Figure 23). The device was connected to an ESP32 to send the collected raw data directly to the ASHVIN database or save it on a memory card. The sensors gathered data during two deliveries of concrete by trucks. The crane was

used to pour concrete into the formwork on the 7th floor after the truck arrived. The activity consisted of four repetitive operations: Lifting bucket down, filling bucket, lifting bucket up, and pouring concrete (Figure 24).



Figure 23 Picture of the device mounted on the crane

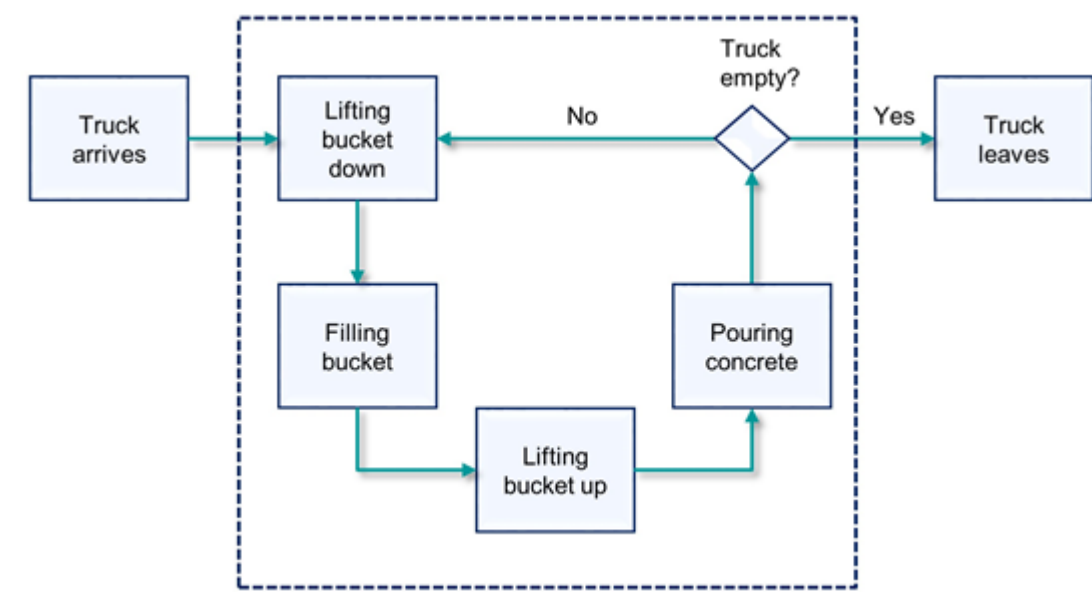


Figure 24 Operations (activities) performed by the crane

For the first truck each operation was repeated nine times and for the second truck eight times besides the operation lifting down, which was repeated only seven times. After the completion of the first truck and the arrival of the second truck there was around one hour of idle time. This time was excluded in the further data mining process.

The different sensors included a three-axis (X/Y/Z) accelerometer, a three-axis gyroscope, a GPS sensor, and a barometer. Thus, it was possible to collect three axes acceleration and angular velocity, longitude/latitude/altitude by the GPS sensor, and height by the barometer. The data from different sensors were fused. During data

collection each data point was saved with a timestamp. Overall, during 98:52 minutes activity time data were collected. This resulted in 27,335 data points. In comparison to other activity recognition studies for construction equipment a low sample rate was chosen, which is usually in the range between 50 Hz and 100 Hz (Sherafat et al. 2020). However, it has been proven in human activity identification that a low sample rate requires less energy and performs well (Zheng et al. 2017).

Data preparation

For analysis of the data R software was used. The collected data were pre-processed before their usage in ML classifiers. Duplicated data points were removed and missing ones were interpolated to finally receive a data set with 4 Hz. Eight data points were merged to create sliding windows of a length of two seconds. The sliding windows were used to calculate six time-domain features: minimum, maximum, interquartile range, mean, variance, and root-mean-square-error. As ten different raw data were collected, the data set contained in total 60 features. The sliding windows were labelled manually by the help of the recorded video.

Classification

Different classifiers were used to compare the performance according to a 10-fold cross validation. The following classifiers were investigated: Naïve Bayes, Decision Tree, k-nearest neighbour (KNN), support vector machine (SVM), and Random Forest. The folds were fixed for testing the classifiers to ensure the same circumstances for each algorithm, although the cross validation is not necessary for the Random Forest classifier as it reduces the possibility of over fitting already. The overall accuracy of each classifier is presented in Table 1. The Random Forest classifier performs by far at best. The overall accuracy is more than 7 % higher than the SVM result as the second-best classifier. For the remaining three classifiers the distance is between 10 and 20 % to the Random Forest performance. As the Random Forest is an advanced version of the Decision Tree, the results are reasonable.

Table 1: Classifier accuracy for 10-fold cross validation

	Naïve Bayes	Decision Tree	KNN	SVM	Random Forest
Accuracy (%)	81.15	87.17	76.84	90.12	97.72

The confusion matrix helps to investigate the performance of classifiers as it compares the actual and the predicted classes. The resulting confusion matrix for the Random Forest classifier is presented in Table 2. As the data set is balanced as each operation has a similar number of instances, the performance of Random Forest seems to be an adequate result. It can be detected that no wrong predictions among the labels Concrete pouring and Filling occur. The reason for this is the different altitude. The wrong predictions occurred in conjunction for the labels Lifting down and Lifting up. The transition between the operations was fluently as e.g. already concrete was poured into the formwork while the bucket was still lifted up.

Table 2: Confusion matrix for Random Forest

		Prediction				Total
		Concrete pouring	Filling	Lifting down	Lifting up	
Actual	Concrete Pouring	1,420	0	4	15	1,439 (24.26 %)
	Filling	0	1,265	9	4	1,278 (21.54 %)
	Lifting Down	8	14	1,491	26	1,539 (25.94%)
	Lifting Up	22	18	15	1,621	1,676 (28.25%)
Total						5,932

The confusion matrix can be used to calculate Precision and Recall for each different operation. These two are performance metrics for classifiers. Precision is the relation of the number of true positives to the total number of positive predicted instances. Recall can be defined as the number of true positives divided by the total number of positives in the data set. In ML models there is a trade-off between Precision and Recall.

Table 3: Precision and Recall for the Random Forest classifier

Operation	Precision [%]	Recall [%]
Concrete Pouring	97.93	98.68
Filling	97.53	98.98
Lifting Down	98.16	96.88
Lifting Up	97.30	96.72

The classified data set resulting from the Random Forest classifier was revised by heuristics. If there were three or less classified data points between series of five or more same instances, the instances were aligned. If there was a single instance between two different classified data points, it was removed. Formally this can be described as:

- Instances Label $x \geq 5$; Instances Label $y \leq 3$; Instances Label $x \geq 5 \rightarrow$ merge all instances to label x
- Instances Label $x \geq 1$; Instances Label $y \leq 1$; Instances Label $z \geq 1 \rightarrow$ remove instance with label y

Afterwards, if there were series of instances ≤ 3 , these instances were removed. The final number of windows was multiplied by one, as the windows have a duration of two seconds with an overlap of 50 % among the windows.

Thus, the duration for each repetition of each operation was calculated as presented in Table 4. In the first row the durations of the first truck are listed and in the second row the durations during the second truck.

Table 4: Resulting operation durations

Operation	Duration [s]
Filling	36, 103, 77, 73, 81, 92, 91, 73, 90; 131, 65, 58, 72, 62, 68, 84, 49
Lifting up	124, 82, 114, 99, 111, 104, 119, 93, 43; 99, 63, 85, 95, 98, 123, 79, 90
Concrete pouring	37, 112, 58, 59, 57, 68, 83, 50, 240; 89, 33, 30, 65, 42, 26, 337, 72
Lifting down	90, 108, 91, 96, 76, 98, 101, 70, 108; 84, 76, 99, 110, 99, 101, 105

These extracted activity durations can be used afterwards in the ASHVIN tools such as the DES tool for planning of upcoming construction works.

4.2.3 Fusion of sensor data for water levels correlation

For PUC10, two types of processes were followed including data collection and data analysis/fusion.

Data was collected from two independent sources including fiber optic sensor (see Figure 25) readings that were converted into a meaningful unit of measurement (height of water), and an online source; <https://waterinfo.rws.nl/>. From the online data source, only the water level height of the water closest to the Quay wall (Figure 26) were selected for this use case. Multiple locations were selected to reduce the possibility of a defective sensor and to avoid moments of noisy measurements.



Figure 11 Positions of fibre optic sensors of PUC10 Quay wall



Figure 26 marks the spot of the locations that were selected (PUC 10).

After the data was obtained, some pre-processing was required. Initially, the data from the website contained information that was irrelevant to the analysis and had to be excluded. In general, the following actions were taken: After eliminating the unused elements, the data had to be transformed into a format that was easier to pre-process and analyse. They were then resampled to a more convenient time frame (measured every one hour) in order to match the measurements with the data provided. Finally, the average value of the water level was collected to provide a more accurate estimate of the water level in the general area of the sensors. The end result of the online data was saved in one file containing all the measurements.

With the data provided, a similar strategy was followed. The data were first checked for missing values before being resampled to remove timestamps that did not match those in the online data file. The initial data analysis revealed that several outliers needed to be removed, as well as various statistical transformations needed to be made to correct measurement errors.

Figure 27 and Figure 28 show the distribution of the measurements.

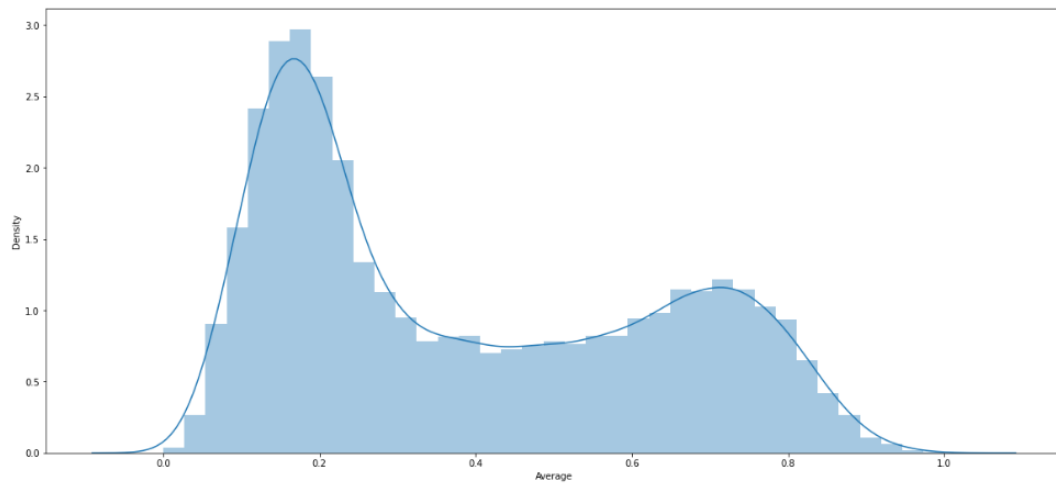


Figure 12 Distribution of PUC10 online data after pre-processing

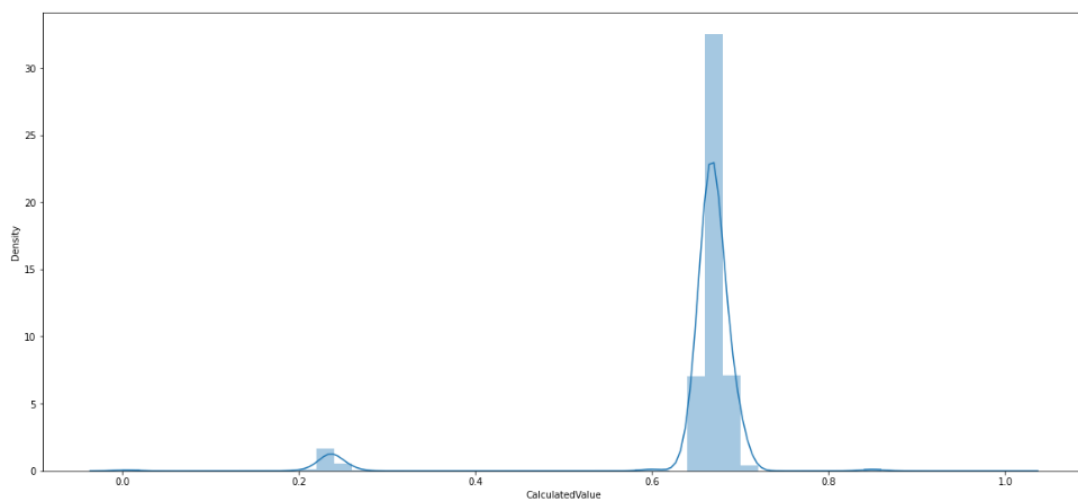


Figure 28 Distribution of PUC10 data after pre-processing

The pipeline that was followed is depicted in Figure 29

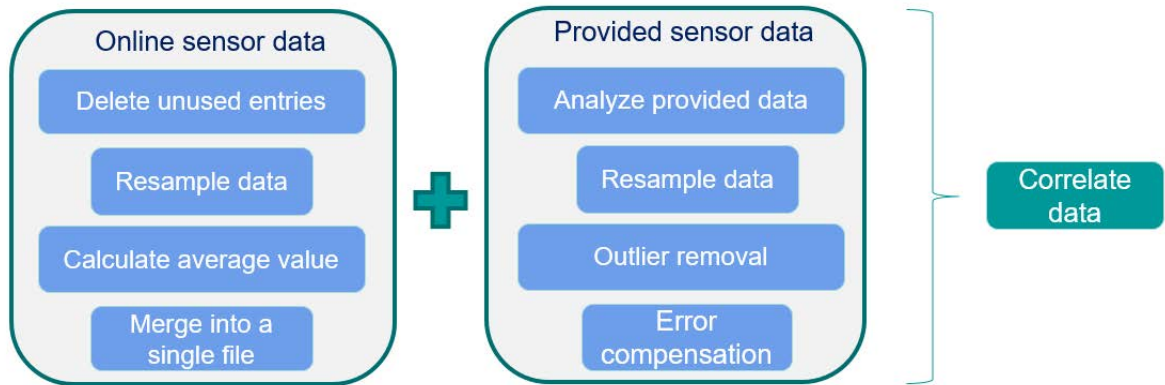


Figure 29 Processing Pipeline for PUC 10

Results

In order to get the best possible correlation between the data, Yeo Johnson transformation was applied. This transformation is used to make the data approach a normal distribution. Various correlation tests were performed and the results are presented in Table 5. The correlation between the measurements is considered weak with the highest value being 0.19 using Spearman's method.

Table 5. Correlation results

	Current Correlation
Kendall	0.12
Spearman	0.19
Pearson	0.03

5. CONCLUSIONS

This report describes Deliverable 3.3 of the Ashvin project related to Task 3.2 IoT data processing and Task 3.4 Data fusion. Several developments and implementations were conducted in order to achieve the goals of these tasks which are to enable IoT data processing of secure real-time data stream pipelines and the analysis and fusion of heterogeneous data collected from pilot sites.

- i) Secure certificate storage and creation with Vault encryption management system
- ii) Utilization of Keycloak for Single sign-on and user management
- iii) Open weather data gathering, processing and aggregating with Telegraf
- iv) Ory Keto implementation for User and Thing groups authorization policy management
- v) Migration from Digital Ocean to AWS Kubernetes and automated DB management with AWS and Velero
- vi) Indoor environmental data gathering, and aggregation with ChirpStack LoRaWAN Network

Regarding the data fusion task, the conclusions from the current applications and progress can be summarized in the following Occupancy detection using environmental sensors is a trending task that also provides room for research, however, a significant amount of sensor readings along with ground truth values, are needed, in order to provide efficient prediction models. It is important to note, that at this phase of the project not all pilot sites have completed the data collection process, thus resulting in some poor performing models.

The results that derived from the analysis of the data provided some basic machine learning models that can be further trained and explored with more data in order to become more reliable. With the reliability of the models increased accurate predictions can be produced.

Future goals for the development of this task are the construction of predictive models based on a selection of the existing sensors and distinguishing the ones that can produce the most reliable models. More fusion methods will be implemented, mainly on research level, as their usage on the exact pilot site may not be effective.

6. REFERENCES

Baldominos, A., Cervantes, A., Saez, Y., and Isasi, P. 2019. "A Comparison of Machine Learning and Deep Learning Techniques for Activity recognition using Mobile Devices". *Sensors* 19(3):521.

Kim, J., Chi, S., and C. R. Ahn. 2021. "Hybrid kinematic-visual sensing approach for activity recognition of construction equipment". *Journal of Building Engineering* 44:102709.

Langroodi, A. K., Vahdatikhaki, F., and A. Dorée. 2021. "Activity recognition of construction equipment using fractional random forest". *Automation in Construction* 122:103465.

Rashid, K. M., and Louis, J. 2019. "Time-series data augmentation and deep learning for construction equipment activity recognition". *Advanced Engineering Informatics* 42:100944.

Sherafat, B., Rashidi, A., Lee, Y.-C., and Ahn, C. R. 2019. "A Hybrid Kinematic-Acoustic System for Automated Activity Detection of Construction Equipment". *Sensors* 19(19): 4286.

Sherafat, B., Ahn, C. R., Akhavian, R., Behzadan, A. H., Golparvar-Fard, M., Kim, H., Lee, Y.-C., Rashidi, A., and E. R. Azar. 2020. "Automated Methods for Activity Recognition of Construction Workers and Equipment: State-of-the-Art Review". *Journal of Construction Engineering and Management* 146(6):03120002.

Slaton, T., Hernandez, C., and R. Akhavian. 2020. "Construction activity recognition with convolutional recurrent networks". *Automation in Construction* 113: 103138.

Zheng, L., Wu, D., Ruan, X., Wenig, S., Peng, A., Tang, B., Lu, H., Shi, H., and H. Zheng. 2017. "A Novel Energy-Efficient Approach for Human Activity Recognition". *Sensors* 17(9):2064.