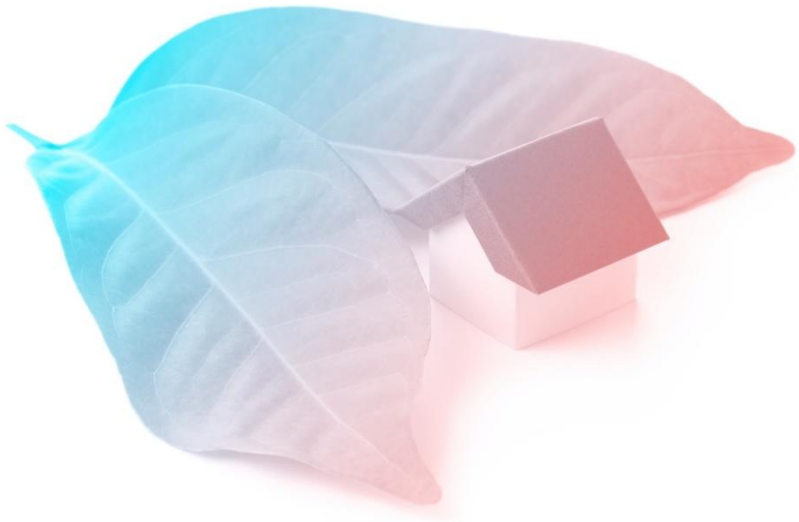
D5.5 Detailed design of the IoT platform for user interaction



Authors:

Nikolaos Tarsounas (CER), Aris Mystakidis (CER), Paschalis Gkadatzis (CER), Dimosthenis Ioannidis (CER), Zoltán Kővári (WOO), Zoltán Pásztory (WOO), Philipp Schütz (HSLU), Susana Gutiérrez (CAR)

This project has received funding from the European Union´s Horizon 2020 research and innovation programme under the grant agreement No 869821

D5.5 Detailed design of the IoT platform for user interaction

Summary

The main purpose of this deliverable is to present the detailed final approach of the user interface of the IoT platform for MiniStor's asset monitoring and control. The main goal was to develop an enhanced user experience and optimize the energy management system. Following activities in task T5.3, a framework is created to allow for the monitoring of critical system parameters and real-time optimization of the system's usage. In this context, this deliverable presents in detail the final approach of the deployed techniques and specifications of the user interface of the IoT platform. It also details the implementation methodology for functionalities and operation modes it offers to the user, the architecture of the data layer (communication platform) as well as documentation for the REST API and monitoring device variables. System parameters are described in respect to each pilot site as there are certain differences, which occur due to the diverse type of monitoring equipment and communication protocols present at the pilot sites. Finally, the document describes the test plans and methodologies for integration and acceptance of the system's components as well as a list of the integrated software and hardware components.

Deliverable Number	Work Package
D5.5	WP. 5
Lead Beneficiary	**Deliverable Author(S)**
Centre for Research and Technology – Hellas CERTH	Nikolaos Tarsounas (CER), Aris Mystakidis (CER), Paschalis Gkaidatzis (CER), Dimosthenis Ioannidis (CER), Zoltán Kővári (WOO), Zoltán Pásztory (WOO), Philipp Schütz (HSLU), Roberto Arnanz (CAR), Luis V, Jimeno (CAR), Carlos Ochoa (IER).
Lead Beneficiary	**Deliverable Reviewer(S)**
IERC	Carlos Ochoa
CARTIF	Alberto Belda
WOODSPRING	Zoltan Kovari
Planned Delivery Date	**Actual Delivery Date**
30/06/2025	29/08/2025

Type of deliverable	R	Report	X
Dissemination Level	CO	Confidential, only for members of the consortium (including the Commission)	
	PU	Public	X

Index

List of Tables

List of Images

List of Abbreviations and Acronyms

The following abbreviations will be frequently used:

API	Application Programming Interface
BASH	Bourne Again Shell
CSV	Comma Separated Values
DR	Demand Response
HVAC	Heating, Ventilation and Air Conditioning
IoT	Internet of Things
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
REST	Representational State Transfer
RPi	Raspberry Pi
SQL	Structured Query Language
UI	User Interface
XML	Extensible Markup Language

1 Introduction

1.1 Scope and objectives of the deliverable

This deliverable presents the work developed in the context of Task 5.3 "IoT-platform for user interaction with system for operation and performance" and is the final deliverable of a total of two deliverables. This deliverable includes a description of the final version of the user interface of the IoT platform, as well as modules needed to support its function and that were developed from M04 until M19 and integrated during the last active period of those tasks, i.e. M57-M68. In addition, this document can serve as a guide for the REST API that is used to manage data sources used by the platform. This report also documents work done for Task 5.4 "IoT- HEMS platform prototyping and acceptance tests", since both tasks are closely correlated but there is no deliverable associated to T5.4.

1.2 Structure of the deliverable

The present document is composed of six chapters that cover:

- Chapter 1 - Introduction: aims to introduce the scope of the deliverable and to define relations with other tasks and deliverables.
- Chapter 2 - Design Approach & Methodology of the IoT platform: presents the final development status of the user interface of the IoT-platform.
- Chapter 3 - IoT Data Layer Design & Implementation: provides details about the data storage components and their structure and outlines the API endpoint documentation.
- Chapter 4 - IoT User Interface Design & Implementation: Functionalities of the interface and its graphic design.
- Chapter 5 - Testing Methodology: analysis of the testing plans to assist the integration and acceptance of the systems components.
- Chapter 6 - Components Integrated with the IoT platform: presents a list of the use case scenarios and gateway development.

1.3 Relation to Other Tasks and Deliverables

This deliverable is a product of Task 5.3, which is part of WP5 "Automated MiniStor Self-Optimization and Control Management Platform". Also, the deliverable includes results from the Task 5.4 as it is closely correlated to the actions taken in Task 5.3 and also there is not a dedicated deliverable for this task. The task 5.3 takes input from the Task 7.1 "MiniStor design evaluation and user behavior validation through mock-ups and user stories" that defines a mock-up user interface based on user requirements. The user interface is developed in accordance to this mock-up that can be found in D7.1 [1]. Also, input is taken from Task 5.1 "Design of the MiniStor control and self-optimization platform" as well. Specifically, the control strategies that are going to be used for the automatic usage and optimization of the system are going to be integrated with the front-end interface. Also, the KPI formulas that are visualized in the platform have been defined in Task 6.1 "Design of the monitoring, definition of KPIs and design of remote data access" and its corresponding deliverable. This deliverable serves as the latest updated version of the previous deliverable, i.e. deliverable D5.4, where the initial design of the IoT platform has been presented.

2 Design Approach & Methodology of the IoT platform

This chapter describes the methodology and the approach followed to reach the final framework architecture of the system. Note that the architecture described in this deliverable is the final iteration out of a total of two. During the life span of the MiniStor project, validation of the system took place in a continuous and iterative manner to optimize interfaces, procedures and components for improved user experience. This iteration of the deliverable provides a complete version of the user interface with all the required systems and components developed from other technical work packages.

2.1 Methodology

The development of the IoT platform continued as stated in deliverable D5.4, i.e. by defining the user and business requirements for it, taking into account a state-of-the-art analysis. Using these requirements, we defined the conceptual architecture, a high-level view of the overall system, which later led to the creation of the system's static and dynamic view. As a last step we specified detailed architectural elements, like identifying inputs and outputs, data exchanges, etc. The IoT platform was developed following the Scrum methodology [2].

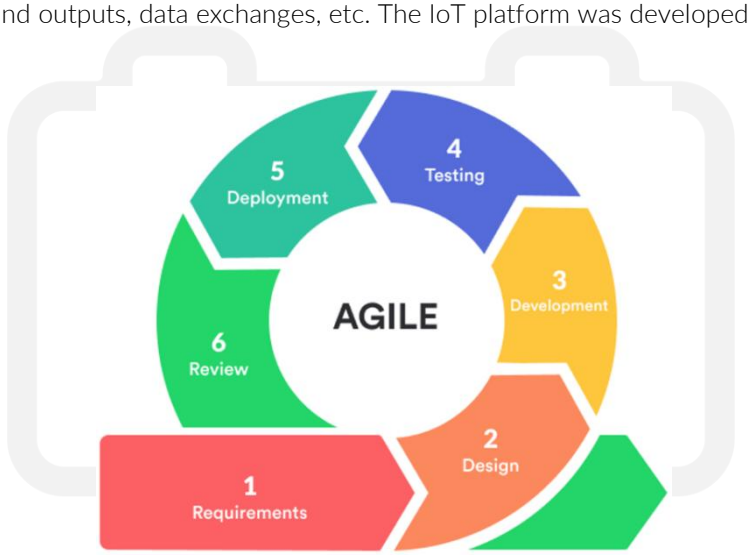


Figure 1 Agile Methodology Steps

Scrum is an Agile [3] based methodology for development. As it is usual in Agile methodologies, the project was divided into small pieces and built incrementally, while ensuring the ability to adapt and change in any step of the process. In Scrum the team decides on how to implement each task and the client only specifies what the outcomes should be. Scrum follows a sprints format as Agile (Figure 1). A sprint is "a set period of time during which specific work has to be completed and made ready for review". So, each of these sprints is time allocated to a few specific tasks and deliverables, but in the end of the sprint the item made must be shippable (e.g. to the final user or other work groups). During a sprint, team members have daily short meetings to discuss their progress with the team manager and the project owner. While the team manager oversees the team, he does not give tasks to the team members directly. His role is to coach the team members so they can deliver high quality products. The project owner's is there to represent the vision of the client and set the sprints priorities. At the end of the sprint the team along with the team manager and the project owner discuss on improvement points for the next sprint. A time schedule for this task that follows this approach can be seen in Figure 2.

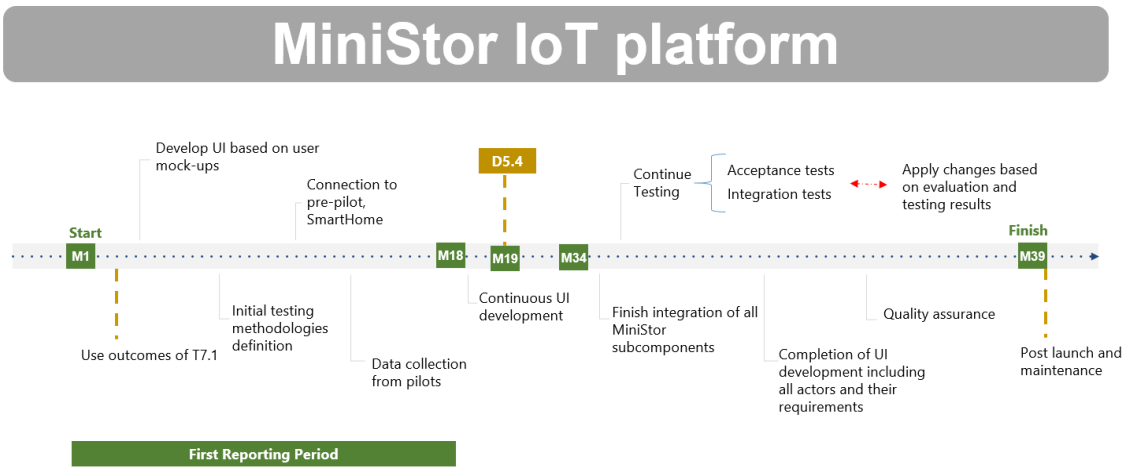


Figure 2 MiniStor IoT User Interface Development Schedule

2.2 Context & Design

The design of the user interface of the IoT platform is based on use cases and user requirements that arise from the user interaction with the MiniStor system, which were collected during Task 7.1. The basic functionalities that resulted from this analysis include:

i. Monitoring of the system's operation and performance (historical, current and future data) as well as the indoor and outdoor environmental conditions
ii. Management of the system's usage by transmitting actions to the system or by utilizing the automatic self-optimization component that utilizes forecasting modes powered by machine learning to control the system efficiently
iii. Monitoring of the Demand-Response (DR) events status, their effectiveness and manual scheduling of the system's usage
iv. Notification system that notifies users in case of faulty equipment or when conditions set by the user are met, and personalization of the user interface based on user preferences

The main purpose of the platform is to allow users to live in a comfortable environment while saving money by improving the energy efficiency of their dwellings. This can be achieved by optimizing the day-to-day usage of the system in order to avoid unnecessary energy-consuming actions in the dwelling with the overall objective of saving money from their energy bills without affecting user wishes. To achieve maximum savings, a user can utilize the platform as described in the following scenarios:

i. Identify potential patterns in the electricity or thermal energy consumption that allows tuning their device usage accordingly in order to reduce costs without hindering quality of life
ii. Enable the self-optimization module in order let the system tune its usage automatically, again targeting maximum saving without any change in the perceived usage
iii. Enable schedules for the system (e.g., thermal) to turn on just before the arrival of the user to the building (e.g., return from work)
iv. Manage DR Events for selling excess energy to the grid based on current and predicted conditions, while being aware of the sell prices
v. Manage DR Events for controlling the need for energy and battery balances

Note that Demand-Response is a procedure that aims to propagate signals to the users for specific time periods to adjust their power usage to better match the supply. This allows them to reduce or shift their

power usage during peak hours to reduce their financial costs. For the user types, there are two main ones, the basic user and the admin. The difference between those two users are mainly their privileges. In addition to information available to a normal user, basically, an admin can check and handle the MiniStor infrastructures, like for example to add or remove buildings or check notifications about the system's functional state. More information can be accessed in the relevant deliverable where basic types of users are described in more detail (D7.1).

The architecture of the IoT platform is based on the client-server approach, where the client (front-end) is basically the user interface that allows user interaction, and the server (back-end) provides the required services, data and management of requests for the front-end functions to work. The user interface contains six main tabs, which are: i) MiniStor Pilot (see section 4.1.2), ii) Data Analytics (see section 4.1.3), iii) Control Panel (see section 4.1.4), iv) Prediction (see section 4.1.5), v) DR Events (see section 4.1.6), and vi) MiniStor System (see section 4.1.7). The back-end utilizes a REST API as a communication vessel to interact in a controlled manner with the databases and the provided web services (HEMS platform).

2.3 Approach and Technologies

The structure of the IoT platform follows a central focused approach, where all IoT components interact with it via a REST API. The user interface (UI) of the platform is implemented with Angular[1], while the back-end is written on Node.js[2].

2.3.1 IoT Platform Architecture

The data monitoring process involves the propagation of data from the data transmission devices of each demo site, after accumulating all the necessary data locally on gateways, to the respectful databases through a REST API (Figure 3). Then, the user interface (UI) of the IoT platform retrieves the data to create intuitive plots and optimize the operation of the system. The API handles the data in JavaScript Object Notation (JSON) format, which is a standard and human-readable file format that is generally used for server communication. The JSON format consists of key-value pairs or key-serializable value pairs for more complex structures.

[1] https://angularjs.org/
[2] https://nodejs.org/en/

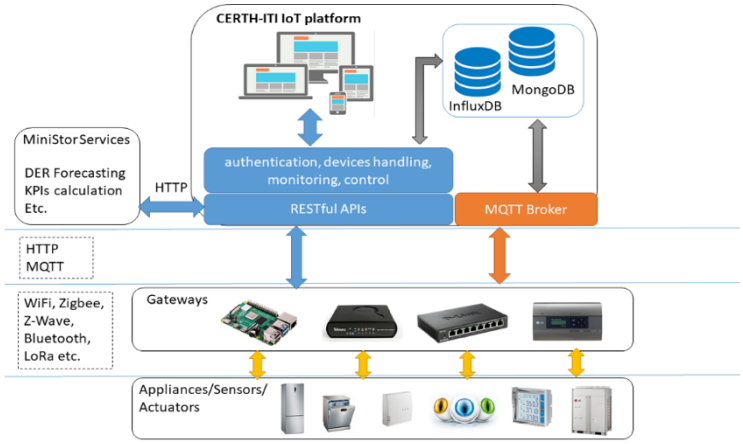


Figure 3 IoT platform ecosystem

2.3.2 Development Technologies

Angular is an open-source framework that is designed to support the display of dynamic content, which is a major limitation of plain HTML. This is achieved by the two-way data binding between a UI element and its corresponding data model. This means that whenever there is a change in both elements, the other is updated accordingly, making it ideal for single-page web pages. The logic of the UI elements is defined in JavaScript, a widely used web-development language.

Angular offers a set of basic functions that are natively pre-build, while at the same time allowing the programmer to create custom ones to capture more advanced scenarios. Also, custom functions are implemented by community members, which further enriches the functionality of the language. Those features make Angular a versatile framework with strong performance.

The representational state transfer (REST) is an application programming interface (API) that is used as a means of communication with other REST web services. It is a software distributed hypermedia system, and it aims to empower data transferring over the web in a simple, secure, scalable, reliable and portable way with good performance. Node.js is a lightweight cross-platform and open-source environment written in JavaScript that offers fast performance. Generally, Node.js is a widely used technology and it is usually used to develop web servers for distributed applications.

3 IoT Data Layer Design & Implementation

This chapter describes the design and the architecture of the data layer as well as the principles followed to create a secure environment. For the needs of the MiniStor project, a REST API is utilized that handles the system data as data needs to be propagated through the IoT platform as well as the HEMS platform. The REST API derives data from two databases. In the following sections, the data handling system is described.

3.1 Data Storage and Security

Data acquired using the MiniStor system are collected and handled in accordance to the EU's and each demo site's national legislation, including GDPR for personal data protection where it applies. More information on the data management plan is described in D1.7 [4].

Data security is ensured by following approaches that comply with the properties of information security, which are data availability, integrity and confidentiality [5]. The sensor data collection is performed securely over the Hypertext Transfer Protocol Secure (HTTPS) that utilizes an encrypted communication protocol. In addition, the data collection in each demo site is monitored in order to ensure the correct operation of the process and the integrity of the data. Also, multi-tenancy is supported, which means that tenants have access only to their own data. These properties are achieved by the utilization of a secure REST API that allows access to the data through encrypted user authentication processes, which promotes data anonymity as well through controlled access.

The aforementioned REST API is connected to two databases. One database is used to save device, user and site details (MongoDB) and the second is used to save the monitoring data of the devices (InfluxDB). The structure of the databases is developed to support the differences of the pilot sites' residency. Access to the data can be achieved either through the user interface of the IoT platform or through code programming. The data can be accessed directly through the databases, but this method is not available publicly, as it is not good practice due to exposure to security risks. Such security risks could be data access by unauthorized personnel or deletion of data. Instead, communication is performed through specific endpoints of a REST API that requires user authentication to enhance safety and privacy.

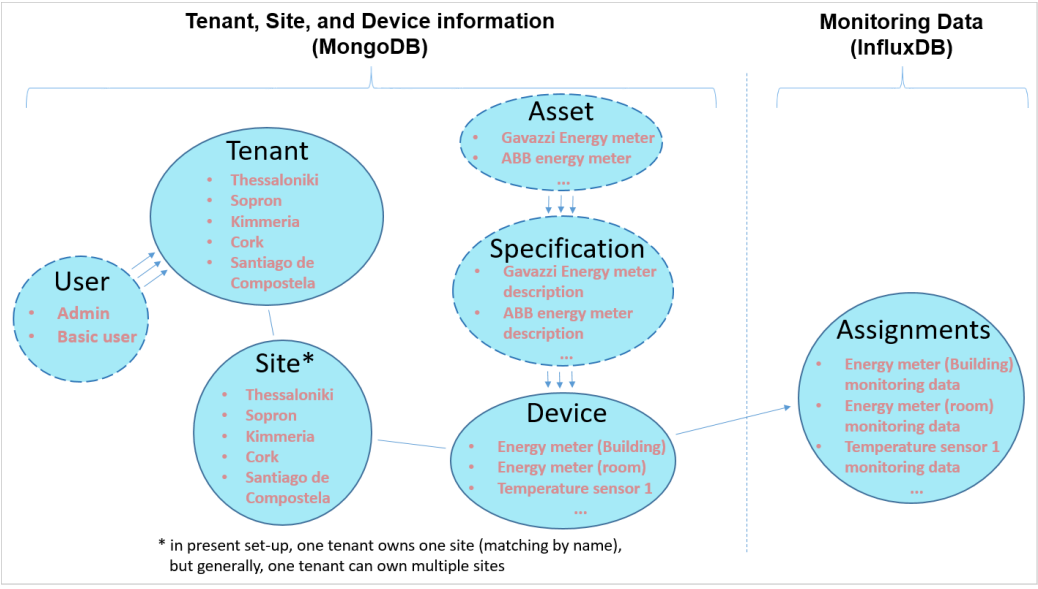


Figure 4 Main Endpoints of the IoT's platform Rest API

The API offers multiple endpoints, but the main ones are the site, user, tenant, assignments, device, specification and asset (Figure 4). The site endpoint allows interaction with information of the pilot sites' location. The user endpoint defines the type of the user and thus his privileges (e.g. basic user, admin), while the tenant manages one or more pilot sites. The asset and specification endpoints are used for the descriptive properties of the devices, while the device endpoint is used to assign a token to a device that is used as device id for monitoring data endpoint (assignments). Finally, the assignments endpoint provides the upload and fetch functions for monitoring data.

3.2 Documentation of API operations and endpoints

In our setting, the API allows for posting (POST) and retrieving (GET) monitoring data, while for the user, site, and device information, specifically, it further offers the ability to alter (PUT) or delete them (DELETE) (Figure 5). The API endpoint list follows.

Figure 5 REST API supported functions

3.2.1 Endpoints for handling *assets*

Assets (Figure 6) define the kind of the device and its name. For example, a kind of device could be an electrical energy meter, whose (trademark) name is Carlo Gavacci.

assets : Operations related to IoT assets. Show/Hide | List Operations | Expand Operations | Raw

GET	/assets/categories	List asset categories that match criteria
POST	/assets/categories	Create a new asset category
GET	/assets/categories/{categoryId}	Get an asset category by unique id
DELETE	/assets/categories/{categoryId}	Delete an existing asset category
PUT	/assets/categories/{categoryId}	Update an existing asset category
GET	/assets/categories/{categoryId}/assets	List category assets that match criteria
GET	/assets/categories/{categoryId}/assets/{assetId}	Get a category asset by unique id
DELETE	/assets/categories/{categoryId}/assets/{assetId}	Delete an existing category asset
POST	/assets/categories/{categoryId}/hardware	Create a new hardware asset in category
PUT	/assets/categories/{categoryId}/hardware/{assetId}	Update an existing hardware asset in category
POST	/assets/categories/{categoryId}/locations	Create a new location asset in category
PUT	/assets/categories/{categoryId}/locations/{assetId}	Update an existing location asset in category
POST	/assets/categories/{categoryId}/persons	Create a new person asset in category
PUT	/assets/categories/{categoryId}/persons/{assetId}	Update an existing person asset in category

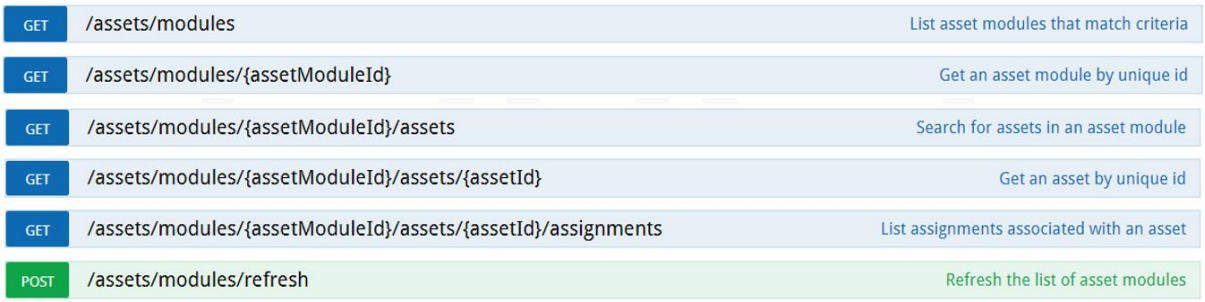


Figure 6 Endpoints for handling "assets"

3.2.2 Endpoints for handling *specifications*

The *specification*'s (Figure 7) endpoint can be considered as a child of the *asset* endpoint, which is used to define the format of the measurements that can be drawn from a specific device, as well as some complementary information. Note that the term "child" means that except from its own properties, the *specification*'s endpoint has the same properties of its "parent", which is the *assets* endpoint.

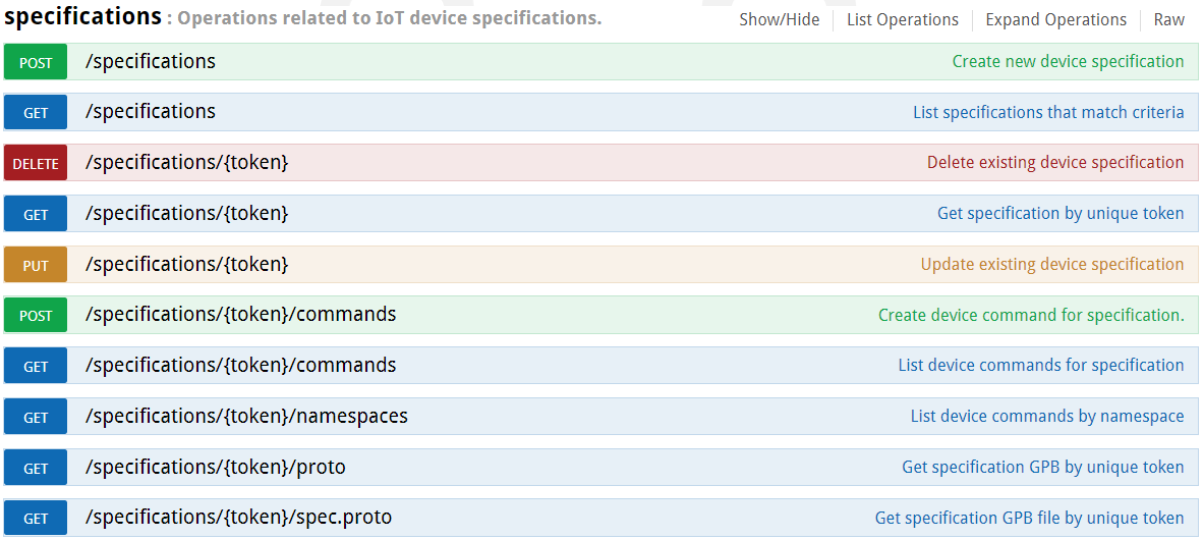


Figure 7 Endpoints for handling "specifications"

3.2.3 Endpoints for handling *devices*

The *device*'s (Figure 8) endpoint is a child of the specifications endpoint, and its use is to assign a unique token to each device that is used for device identification in the different functions offered by the API. This endpoint can be considered as the virtual representation of a physical device.

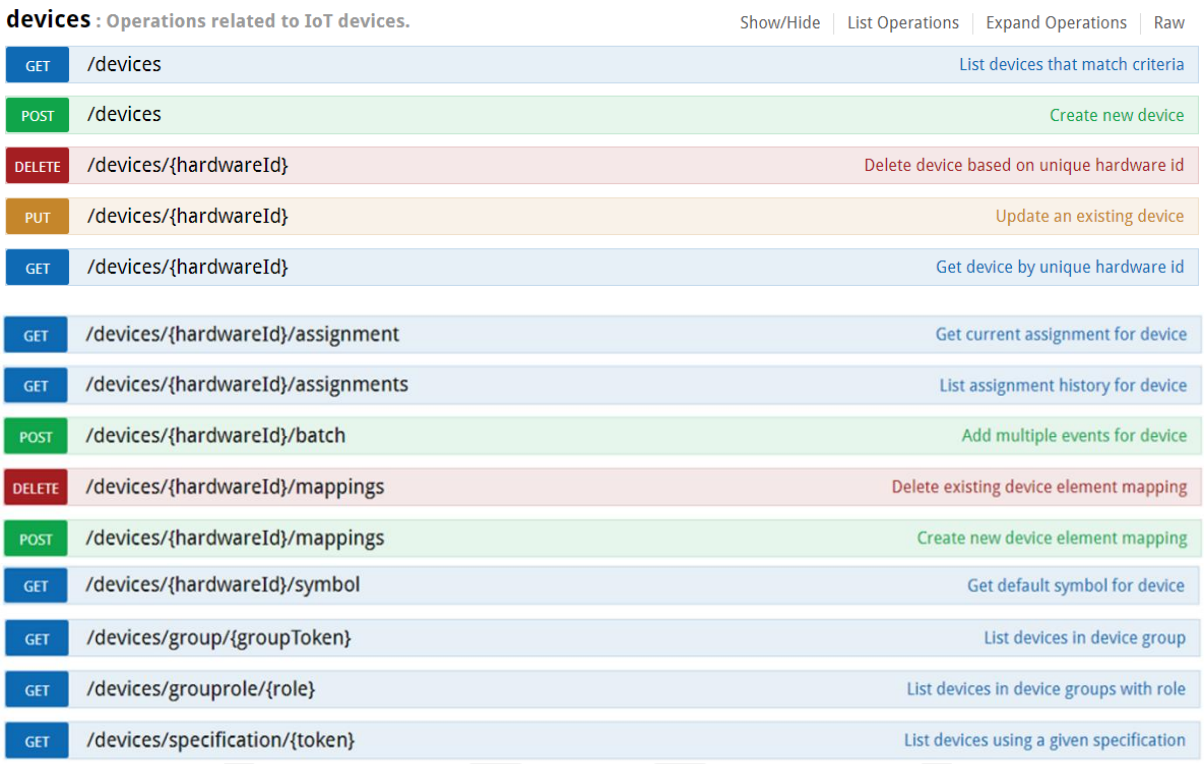


Figure 8 Endpoints for handling "devices"

3.2.4 Endpoints for handling *assignments*

The *assignments* (Figure 9) endpoint provides device data. From this endpoint, monitoring data can be accessed under the path: assignments, device token and then measurements. Data can be retrieved either as time series or just the latest recorded value.

assignments : Operations related to IoT device assignments. Show/Hide | List Operations | Expand Operations | Raw

POST	/assignments	Create a new device assignment
DELETE	/assignments/{token}	Delete an existing device assignment
GET	/assignments/{token}	Get device assignment by token
POST	/assignments/{token}/alerts	Create alert event for device assignment
GET	/assignments/{token}/alerts	List alert events for device assignment
GET	/assignments/{token}/degradation	List measurement events for device assignment
POST	/assignments/{token}/end	Release an active device assignment
GET	/assignments/{token}/events	List events for device assignment
POST	/assignments/{token}/invocations	Create command invocation event for assignment
GET	/assignments/{token}/invocations	List command invocation events for assignment
POST	/assignments/{token}/invocations/schedules/{scheduleToken}	Schedule command invocation
POST	/assignments/{token}/locations	Create location event for device assignment
GET	/assignments/{token}/locations	List location events for device assignment
GET	/assignments/{token}/measurements	List measurement events for device assignment
POST	/assignments/{token}/measurements	Create measurements event for device assignment
GET	/assignments/{token}/measurements/checkSensor	Get last two measurement events for device assignment
GET	/assignments/{token}/measurements/lastValue	Get last measurement events for device assignment
GET	/assignments/{token}/measurements/series	List assignment measurements as chart series
GET	/assignments/{token}/measurementsAggregated	List measurement events for device assignment
PUT	/assignments/{token}/metadata	Update device assignment metadata
POST	/assignments/{token}/missing	Mark device assignment as missing
GET	/assignments/{token}/responses	List command response events for assignment
POST	/assignments/{token}/responses	Create command response event for assignment
GET	/assignments/{token}/statechanges	List state change events for a device assignment
POST	/assignments/{token}/statechanges	Create an state change event for a device assignment
POST	/assignments/{token}/streams	Create data stream for a device assignment
GET	/assignments/{token}/streams	List data streams for device assignment
POST	/assignments/{token}/streams/{streamId}	Add data to device assignment data stream
GET	/assignments/{token}/streams/{streamId}	Get device assignment data stream by id
GET	/assignments/{token}/streams/{streamId}/data	Get all data from device assignment data stream
GET	/assignments/{token}/streams/{streamId}/data/{sequenceNumber}	Get data from device assignment data stream
GET	/assignments/{token}/symbol	Get default symbol for assignment

Figure 9 Endpoints for handling "assignments"

3.2.5 Endpoints for handling *schedules*

The *schedules* (Figure 10) endpoint is used to schedule the time and date that the heating system will be working.

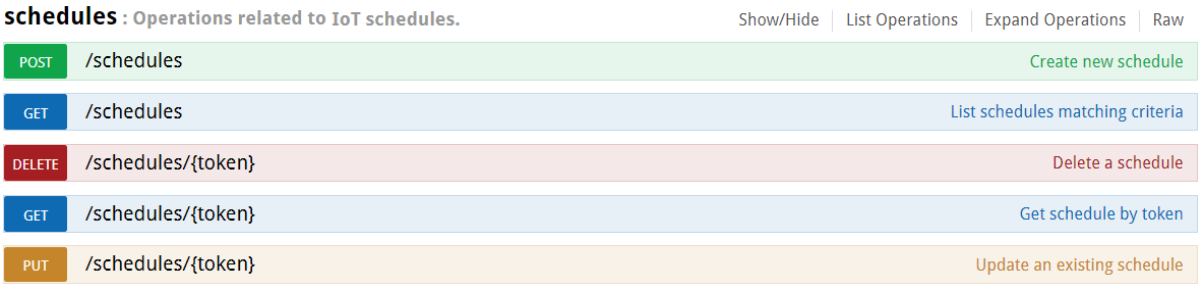


Figure 10 Endpoints for handling "schedules"

3.2.6 Endpoints for handling *notifications*

The *notifications* (Figure 11) endpoint handles the notification system of the platform. Mainly, notifications can be triggered based on conditions set by the user or in order to inform the user of malfunctioning components.

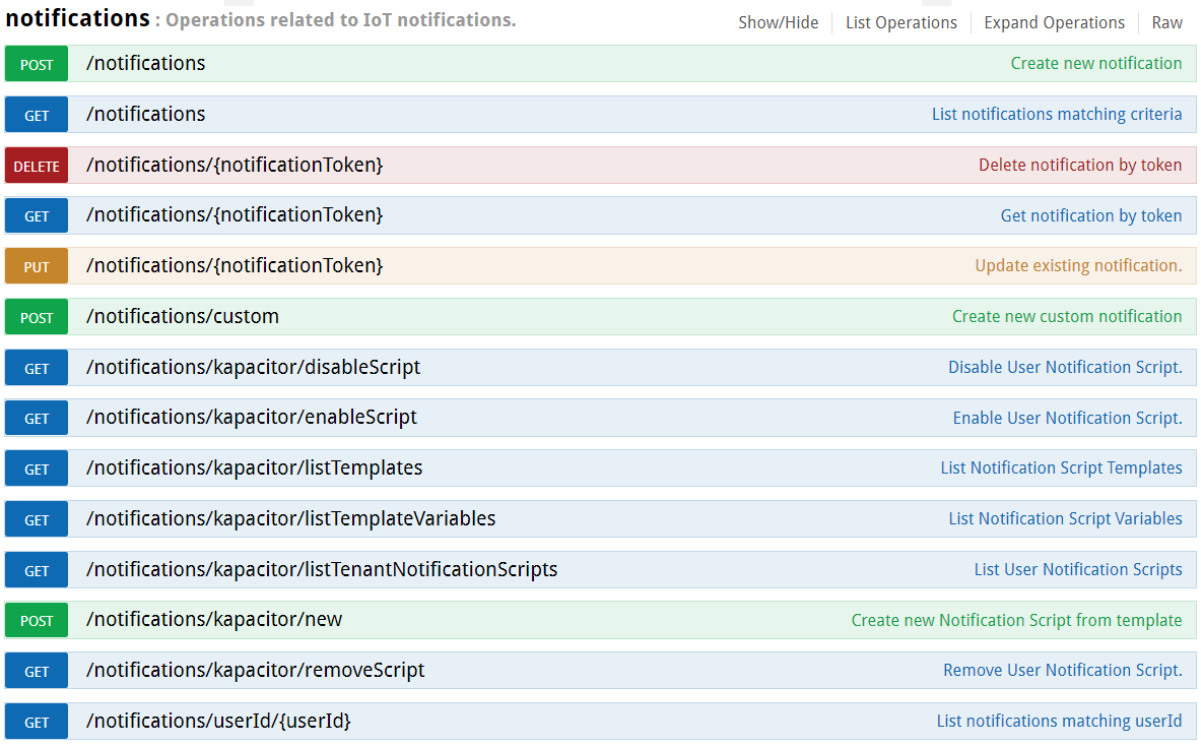


Figure 11 Endpoints for handling "notifications"

3.2.7 Endpoints for handling UI widgets of the *dashboards*

The *dashboards* (Figure 12) endpoint is used to define the menu and the widgets of each tab of the user interface of the IoT platform. This allows for dynamic construction of different layouts for each actor individually so that the platform suits their needs and to further personalize the user's experience.

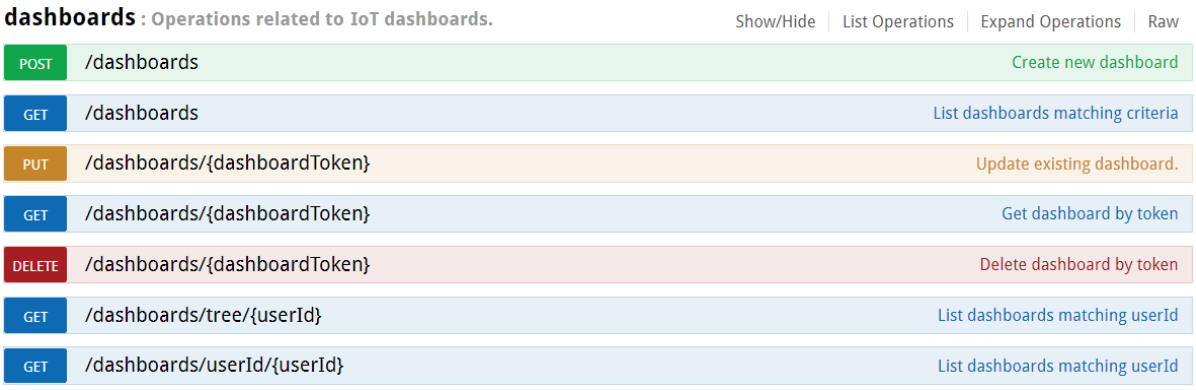


Figure 12 Endpoints for handling "dashboard"

3.2.8 Endpoints for handling *users*

The *user*'s (Figure 13) endpoint is used to define the user type and their privileges (e.g. basic user or administrator).

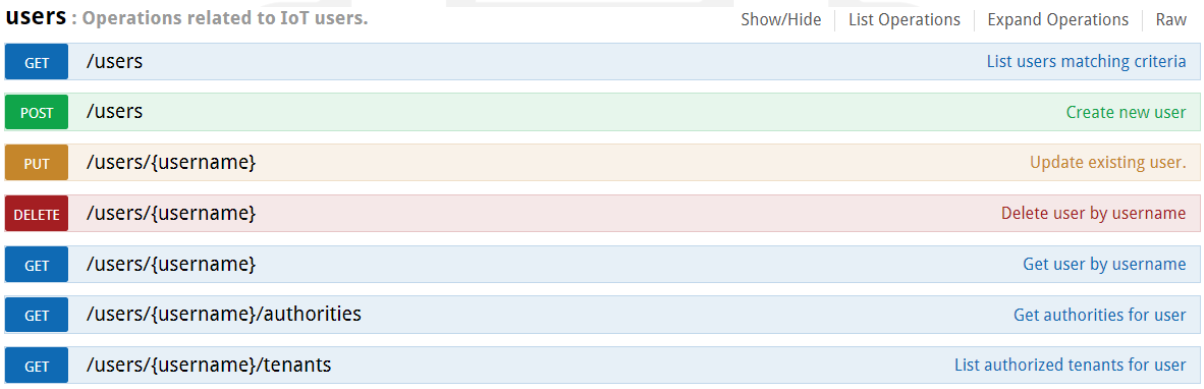


Figure 13 Endpoints for handling "users"

3.2.9 Endpoints for handling *tenants*

The *tenant*'s (Figure 14) endpoint is a child of the user endpoint and contains information about a specific tenant.

tenants : Operations related to IoT tenants. Show/Hide | List Operations | Expand Operations | Raw

POST	/tenants	Create new tenant
GET	/tenants	List tenants that match criteria
PUT	/tenants/{tenantId}	Update an existing tenant.
DELETE	/tenants/{tenantId}	Delete existing tenant
GET	/tenants/{tenantId}	Get tenant by unique id
POST	/tenants/{tenantId}/engine/{command}	Send command to tenant engine
GET	/tenants/{tenantId}/engine/configuration	Get tenant engine configuration
GET	/tenants/{tenantId}/engine/configuration/json	Get tenant engine configuration as JSON
POST	/tenants/{tenantId}/engine/configuration/json	Stage tenant engine configuration from JSON
GET	/tenants/authtoken/{authToken}	Get tenant by authentication token
GET	/tenants/device/{hardwareId}	List tenants that contain a device

Figure 14 Endpoints for handling "tenants"

3.2.10 Endpoints for handling *sites*

The *sites* (Figure 15) endpoint is used to interact with pilot site information as well as some functions tied to a specific pilot site, for example inspection of alerts. Note that one tenant can own or manage multiple sites.

sites : Operations related to IoT sites. Show/Hide | List Operations | Expand Operations | Raw

GET	/sites	List sites matching criteria
POST	/sites	Create new site
DELETE	/sites/{siteToken}	Delete site by unique token
GET	/sites/{siteToken}	Get site by unique token
PUT	/sites/{siteToken}	Update existing site

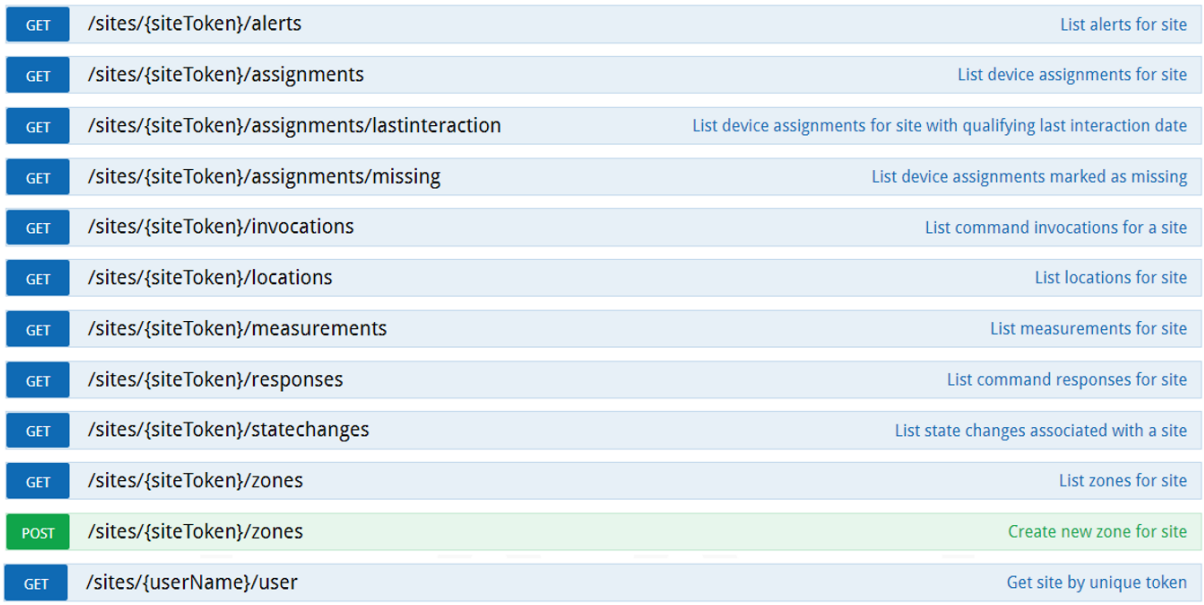


Figure 15 Endpoints for handling "sites"

3.2.11 Endpoints for handling *user preferences*

The *user preferences* (Figure 16) endpoint is used to store the settings of the IoT platform for each tenant, and it mainly contains styling settings of the user interface.

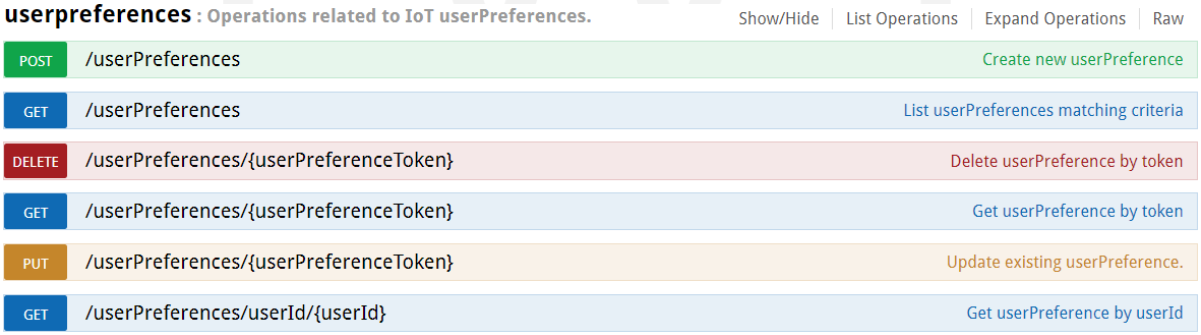


Figure 16 Endpoints for handling "user preferences"

3.3 Device variables

For each demo site, monitoring data are organized with regard to the measuring devices used. Each device accommodates one or more sensors that are able to provide digitized measurement values for certain physical attributes, also called variables. Selection of required measuring devices was performed in Task 6.1, with the goal being the ability to capture all relevant variables that are needed for calculating the Key Performance Indicators (KPIs) of the project. For the definitions of KPIs, as well as for the list of specifications of selected devices for each site, please see 'D6.1 Design of the monitoring system and KPI definition'.

The following tables show indicative definitions of variables used in relation to each type of measurement device. Each row describes one of the possible data fields in the JSON-formatted input or response messages of the REST API described in the previous section. Also, please note that we have listed several variables separately that are not available at every demo site due to differences in capabilities of acquired devices. These variables may not be strictly necessary for KPI calculations, but their values are recorded for supplementary purposes at the request of the responsible partner.

Name	Unit	Cumulative	Aggregate	Site Availability
Temperature	°C	No	No	All
Relative Humidity	%	No	No	All

Table 1 Variables of indoor temperature and humidity sensors (Elvaco CMa10, Elvaco CMa11, Plugwise Sense, S+S RTF1-NTC10k)

Name	Unit	Cumulative	Aggregate	Site Availability
Temperature	°C	No	No	All
Relative Humidity	%	No	No	All
Solar Radiation	W/m^2	No	No	All
Wind Speed	m/s	No	No	All
Wind Direction	°	No	No	All
Dew Point Temperature	°C	No	No	Cork, Sopron
Wind Gust Speed	m/s	No	Yes	All
Wind Gust Direction	°	No	Yes	Cork, Sopron
Barometric Pressure	mBar	No	No	Cork
Inside Temperature	°C	No	No	Cork
Inside Humidity	%	No	No	Cork
Rain Rate	mm/h	No	No	Cork, USC
Rainfall Amount (Storm)	mm	Yes	No	Cork, USC
Rainfall Amount (Day)	mm	Yes	No	Cork, USC
Evapotranspiration (Day)	mm	Yes	No	Cork,
Wet Bulb Temperature	°C	No	No	Cork
Heat Index	°C	No	No	Cork
Wind Chill	°C	No	No	Cork
Temp-Hum-Sun-Wind Index	°C	No	No	Cork
UV Index		No	No	Cork, USC
Sonic Temperature	°C	No	No	Sopron
Absolute Humidity	g/m^3	No	No	Sopron

Table 2 Variables of weather stations (Davis Vantage Pro 2 plus, DeltaOHM HD52.3DP17, FINoT Agri)

Name	Unit	Cumulative	Aggregate	Site Availability
Thermal Energy Total	kWh	Yes	No	Cork, Kimmeria, Sopron
Volume Total	m^3	Yes	No	Cork, Kimmeria, Sopron
Power	W	No	No	Cork, Sopron
Volume Flow	m^3/h	No	No	Cork, Sopron
Supply Temperature	°C	No	No	Cork, Sopron
Return Temperature	°C	No	No	Cork, Sopron
Temperature Difference	K	No	No	Sopron

Table 3 Variables of heat flow meters (BMeters Ultrasonis ULC, GMDM-I + IWM-PL3 + Hydrosplit-M3, Hydrocal M3, Itron CF-UltraMaXX V)

Name	Unit	Cumulative	Aggregate	Site Availability
Gas Volume Total	m^3	Yes	No	Cork

Table 4 Variables of gas flow meter (Honeywell BK-G4)

Name	Unit	Cumulative	Aggregate	Site Availability
Fresh Air Temperature	°C	No	No	Sopron
Supply Air Temperature	°C	No	No	Sopron
Extract Air Temperature	°C	No	No	Sopron
Exhaust Air Temperature	°C	No	No	Sopron
Supply Air Flow	m^3/h	No	No	Sopron
Exhaust Air Flow	m^3/h	No	No	Sopron

Table 5 Variables of HVAC unit (Airvent Microplex)

Name	Unit	Cumulative	Aggregate	Site Availability
Active Energy Total	kWh	Yes	No	All
Amperage	A	No	No	Cork, Sopron, Thessaloniki
Frequency	Hz	No	No	Cork, Sopron, Thessaloniki
Power Factor		No	No	Cork, Sopron, Thessaloniki
Voltage	V	No	No	Cork, Sopron, Thessaloniki
Apparent Power	VA	No	No	Cork, Sopron, Thessaloniki
Reactive Power	VAr	No	No	Cork, Sopron, Thessaloniki
Active Power	W	No	No	Cork, Sopron, Thessaloniki
Active Power Integrated	W	No	Yes	Cork, Sopron
Peak Value of Active Power Int.	W	No	Yes	Cork, Sopron
Reactive Energy Total	kVArh	Yes	No	Cork, Sopron, Thessaloniki
Active Energy Total (Partial)	kWh	Yes	No	Cork, Sopron
Reactive Energy Total (Partial)	kVArg	Yes	No	Cork, Sopron
Active Energy Total (Negative)	kWh	Yes	No	Cork, Sopron
Reactive Energy Total (Negative)	kVArh	Yes	No	Cork, Sopron

Table 6 Variables of 1-phase electrical energy meters (Carlo Gavazzi EM111, Carlo Gavazzi EM112, Circutor CEM C6, Schneider A9MEM2135)

Name	Unit	Cumulative	Aggregate	Site Availability
Active Energy Total	kWh	Yes	No	All
Active Energy Total Line 1	kWh	Yes	No	Sopron
Active Energy Total Line 2	kWh	Yes	No	Sopron
Active Energy Total Line 3	kWh	Yes	No	Sopron
Amperage Line 1	A	No	No	Sopron, Thessaloniki
Amperage Line 2	A	No	No	Sopron, Thessaloniki
Amperage Line 3	A	No	No	Sopron, Thessaloniki
Frequency	Hz	No	No	Sopron, Thessaloniki
Power Factor Line 1		No	No	Sopron, Thessaloniki
Power Factor Line 2		No	No	Sopron, Thessaloniki
Power Factor Line 3		No	No	Sopron, Thessaloniki
Power Factor Total		No	No	Sopron, Thessaloniki
Voltage Line 1	V	No	No	Sopron, Thessaloniki
Voltage Line 2	V	No	No	Sopron, Thessaloniki
Voltage Line 3	V	No	No	Sopron, Thessaloniki
Voltage Line 1 to Line 2	V	No	No	Sopron, Thessaloniki
Voltage Line 2 to Line 3	V	No	No	Sopron, Thessaloniki
Voltage Line 3 to Line 1	V	No	No	Sopron, Thessaloniki
Voltage Line to Line Total	V	No	No	Sopron
Voltage Line to Neutral Total	V	No	No	Sopron
Apparent Power Line 1	VA	No	No	Sopron, Thessaloniki
Apparent Power Line 2	VA	No	No	Sopron, Thessaloniki
Apparent Power Line 3	VA	No	No	Sopron, Thessaloniki
Apparent Power Total	VA	No	No	Sopron, Thessaloniki
Reactive Power Line 1	VAr	No	No	Sopron, Thessaloniki
Reactive Power Line 2	VAr	No	No	Sopron, Thessaloniki
Reactive Power Line 3	VAr	No	No	Sopron, Thessaloniki
Reactive Power Total	VAr	No	No	Sopron, Thessaloniki
Active Power Line 1	W	No	No	Sopron, Thessaloniki
Active Power Line 2	W	No	No	Sopron, Thessaloniki
Active Power Line 3	W	No	No	Sopron, Thessaloniki
Active Power Total	W	No	No	Sopron, Thessaloniki
Active Power Integrated	W	No	Yes	Sopron
Peak Value of Active Power Int.	W	No	Yes	Sopron
Reactive Energy Total	kVArh	Yes	No	Sopron, Thessaloniki
Active Energy Total (Partial)	kWh	Yes	No	Sopron
Reactive Energy Total (Partial)	kVArh	Yes	No	Sopron
Active Energy Total (Negative)	kWh	Yes	No	Sopron
Reactive Energy Total (Negative)	kVArh	Yes	No	Sopron

Table 7 Variables of 3-phase electrical energy meters (Carlo Gavazzi EM340, Circutor CVM-E3-MINI-ITF-485-IC)

Name	Unit	SubSystem	Cumulative	Aggregate	Site Availability
Frigopol Compressor Run	-	TCM	No	No	All
Frigopol Compressor VFD alarm	-	Alarms	No	No	All
Oil thermal safety switch	-	TCM	No	No	All
Emergency Stop Switch	-	TCM	No	No	All
Local ON/OFF switch for TCM	-	TCM	No	No	All
Oil Solenoid	-	TCM	No	No	All
Ammonia SubSystem Alarm	-	Alarms	No	No	All
NH3 High Press Safety Switch	-	TCM	No	No	All
NH3 Liquid Level Switch on Separator	-	TCM	No	No	All
Suction Solenoid Valve	-	TCM	No	No	All
Suction separator Solenoid Valve	-	TCM	No	No	All
Hot Gas Solenoid Valve	-	TCM	No	No	All
Injection Solenoid Valve	-	TCM	No	No	All
Liquid Line Solenoid 1	-	TCM	No	No	All
EEV Solenoid	-	TCM	No	No	All
Desuperheater Fan	-	TCM	No	No	All
Ready for charging fully discharged	-	TCM	No	No	All
Ready for discharging fully charged	-	TCM	No	No	All
Stand by ON/OFF	-	TCM	No	No	All
Pressurization/Depresssurization	-	TCM	No	No	All
TCM Event Discharging Discharging ON/OFF	-	TCM	No	No	All
TCM Event Charging Charging ON/OFF	-	TCM	No	No	All
TCM Command to Charge Request to start charging	-	TCM	No	No	All
TCM Command to Discharge request to start discharging	-	TCM	No	No	All
Frigopol Compressor Hz	Hz	TCM	No	No	All
Frigopol Compressor kW	kW	TCM	No	No	All
Frigopol Compressor Ampere	A	TCM	No	No	All
Frigopol Compressor Volt	V	TCM	No	No	All
Liquid Level Sensor %	%	TCM	No	No	All
TCM Pressure bar	bar	TCM	No	No	All
Discharge Pressure bar	bar	TCM	No	No	All
Suction evaporating Pressure bar	bar	TCM	No	No	All
TCM Reactor Temp1 °C	°C	Temperatures	No	No	All
TCM Reactor Temp2 °C	°C	Temperatures	No	No	All
TCM Reactor Temp3 °C	°C	Temperatures	No	No	All
TCM OUT Temp °C	°C	Temperatures	No	No	All
Suction Temp °C	°C	Temperatures	No	No	All
Discharge Temp °C	°C	Temperatures	No	No	All
Separator OUT Temp °C	°C	Temperatures	No	No	All
Receiver IN Temp °C	°C	Temperatures	No	No	All
Receiver OUT Temp °C	°C	Temperatures	No	No	All
Heat Pump Compressor Run	-	HP	No	No	All
Heat Pump Compressor VFD alarm	-	Alarms	No	No	All

High Press Safety Switch	-	HP	No	No	All
Low Press Safety Switch	-	HP	No	No	All
Heat Pump Alarm Status	-	Alarms	No	No	All
HP Run Command	-	HP	No	No	All
Heat Pump Compressor Hz	Hz	HP	No	No	All
Heat Pump Power Consumption kW	kW	HP	No	No	All
Heat Pump Compressor Ampere	A	HP	No	No	All
Heat Pump Compressor Volt	V	HP	No	No	All
Refrigerant Suction Pressure bar	bar	HP	No	No	All
Refrigerant Discharge Pressure bar	bar	HP	No	No	All
Refrigerant flow kg/s	kg/s	HP	No	No	All
Heat Pump Heating Capacity kW	kW	HP	No	No	All
HP Setpoint °C	°C	HP	No	No	All
Remote Emergency Stop Command	-	Other	No	No	All
Local ON/OFF switch	-	Other	No	No	All
Emergency Stop Switch	-	Other	No	No	All
Reset alarm button	-	Other	No	No	All
Ammonia Detection Sensor Level 1	-	TCM	No	No	All
Ammonia Detection Sensor Level 2	-	TCM	No	No	All
Hot Water PCM Controller Command	-	PCM	No	No	All
DHW PCM Controller Command	-	PCM	No	No	All
Smoke Sensor Alarm	-	Alarms	No	No	All
Universal Input 1	-	Other	No	No	All
Universal Input 2	-	Other	No	No	All
Universal Input 3	-	Other	No	No	All
PCM Pump 1	-	Pumps	No	No	All
NH3 Evaporator Pump	-	Pumps	No	No	All
PCM Pump 2	-	Pumps	No	No	All
HP Condenser Pump	-	Pumps	No	No	All
Solar Buffer Pump	-	Pumps	No	No	All
NH3 Condenser Pump	-	Pumps	No	No	All
Fan Coil 1	-	Fan Coils	No	No	All
Fan Coil 2	-	Fan Coils	No	No	All
Buffer Resistors Backup Heater 2kW	-	Solar	No	No	All
NH3 Compartment Exhaust Fan	-	TCM	No	No	All
Fancoil_cold_in_3wv	-	Valves	No	No	All
TCMevaporator_in	-	Valves	No	No	All
PVTtoPRODpump	-	Valves	No	No	All
PVTtoColdPCM	-	Valves	No	No	All
TCMrecirc_in_3wv	-	Valves	No	No	All
Fancoilheat	-	Valves	No	No	All
MANIFOLDreturn	-	Valves	No	No	All
MANIFOLDsupply	-	Valves	No	No	All
PCMHW	-	Valves	No	No	All
PCMDHW	-	Valves	No	No	All
HPevaporator	-	Valves	No	No	All

TCMcondenser_out	-	Valves	No	No	All
TCMin_3wv	-	Valves	No	No	All
TCMcondenser_in	-	Valves	No	No	All
TCMrecirc_out_3wv	-	Valves	No	No	All
Universal Output 1	-	Other	No	No	All
Universal Output 2	-	Other	No	No	All
Universal Output 3	-	Other	No	No	All
Universal Output 4	-	Other	No	No	All
Universal Output 5	-	Other	No	No	All
Real Power Consumption Ph1 kW	kW	Other	No	No	All
Real Power Consumption Ph2 kW	kW	Other	No	No	All
Real Power Consumption Ph3 kW	kW	Other	No	No	All
Real Power Consumption Total kW	kW	Other	No	No	All
Apparent Power Consumption Ph1 kVA	kW	Other	No	No	All
Apparent Power Consumption Ph2 kVA	kW	Other	No	No	All
Apparent Power Consumption Ph3 kVA	kW	Other	No	No	All
Apparent Power Consumption Total kVA	kW	Other	No	No	All
Voltage Ph1 V	V	Other	No	No	All
Voltage Ph2 V	V	Other	No	No	All
Voltage Ph3 V	V	Other	No	No	All
Current Ph1 A	A	Other	No	No	All
Current Ph2 A	A	Other	No	No	All
Current Ph3 A	A	Other	No	No	All
Current Total A	A	Other	No	No	All
Frequency Ph1	Hz	Other	No	No	All
Frequency Ph2	Hz	Other	No	No	All
Frequency Ph3	Hz	Other	No	No	All
Cos Ph1	°	Other	No	No	All
Cos Ph2	°	Other	No	No	All
Cos Ph3	°	Other	No	No	All
HTF Reactor OUT °C	°C	Temperatures	No	No	All
HTF Reactor IN °C	°C	Temperatures	No	No	All
Solar Buffer °C	°C	Temperatures	No	No	All
HTF HP Condenser OUT °C	°C	Temperatures	No	No	All
HTF NH3 Condenser OUT °C	°C	Temperatures	No	No	All
HTF NH3 Evaporator OUT °C	°C	Temperatures	No	No	All
HTF HP Evaporator OUT °C	°C	Temperatures	No	No	All
HTF NH3 Evaporator IN °C	°C	Temperatures	No	No	All
Cold PCM °C	°C	Temperatures	No	No	All
Spare Temp Sensor Inlet °C	°C	Temperatures	No	No	All
MODE TCM UNIT	-	TCM	No	No	All
Temperature offsite DEMO (get API certh)	°C	Temperatures	No	No	All
Period mode (Summer/Winter) from CARTIF	-	Other	No	No	All

State System from CARTIF	-	Other	No	No	-
State TCM from CARTIF	-	Other	No	No	All
State PCM from CARTIF	-	Other	No	No	All
State Demand from CARTIF	-	Other	No	No	All
State Solar Tank from CARTIF	-	Other	No	No	All
Phase A Current	A	Solar	No	No	Cork, Thessaloniki, USC
Phase B Current	A	Solar	No	No	Cork, Thessaloniki, USC
Phase C Current	A	Solar	No	No	Cork, Thessaloniki, USC
Phase Voltage AB	V	Solar	No	No	Cork, Thessaloniki, USC
Phase Voltage BC	V	Solar	No	No	Cork, Thessaloniki, USC
Phase Voltage CA	V	Solar	No	No	Cork, Thessaloniki, USC
Phase Voltage AN	V	Solar	No	No	Cork, Thessaloniki, USC
Phase Voltage BN	V	Solar	No	No	Cork, Thessaloniki, USC
Phase Voltage CN	V	Solar	No	No	Cork, Thessaloniki, USC
AC Power	W	Solar	No	No	Cork, Thessaloniki, USC
AC Apparent Power	VA	Solar	No	No	Cork, Thessaloniki, USC
AC Reactive Power	Var	Solar	No	No	Cork, Thessaloniki, USC
AC Power Factor	-	Solar	No	No	Cork, Thessaloniki, USC
Cabinet Temperature	°C	Solar	No	No	Cork, Thessaloniki, USC
DC Current	A	Solar	No	No	Cork, Thessaloniki, USC
DC Voltage	V	Solar	No	No	Cork, Thessaloniki, USC

DC Power	W	Solar	No	No	Cork, Thessaloniki, USC
DC Current	A	Solar	No	No	Cork, Thessaloniki, USC
DC Voltage	V	Solar	No	No	Cork, Thessaloniki, USC
DC Power	W	Solar	No	No	Cork, Thessaloniki, USC
Setpoint for maximum charge	W	Solar	No	No	Cork, Thessaloniki, USC
Currently available energy as a percent of the capacity rating.	-	Solar	No	No	Thessaloniki, USC
State of charge ChaState minus storage reserve MinRsvPct times capacity rating AhrRtg.	-	Solar	No	No	Thessaloniki, USC
Internal battery voltage.	V	Solar	No	No	Thessaloniki, USC
Charge status of storage device. Enumerated value.	-	Solar	No	No	Thessaloniki, USC
Total Real Power	W	Solar	No	No	-
Watts phase A	W	Solar	No	No	-
Watts phase B	W	Solar	No	No	-
Watts phase C	W	Solar	No	No	-
AC Apparent Power	VA	Solar	No	No	-
Reactive Power	VAR	Solar	No	No	-
Power Factor	-	Solar	No	No	-
Control Unit Run/Stop	-	Solar	No	No	-
Control Unit Mode	-	Solar	No	No	-
Control Circuit 1 Run/Stop	-	Solar	No	No	-
Control Circuit 1: Water heating Fix Setting Temp	°C	Solar	No	No	-
Operation System	-	Solar	No	No	-
Outdoor ambient temperature	°C	Solar	No	No	-
Water Inlet unit temperature	°C	Solar	No	No	-
Water outlet unit temperature	°C	Solar	No	No	-
Unit Power consumption	W	Solar	No	No	-
Tg: Gas Temperature THMg °C	°C	Solar	No	No	-
Tl: Liquid Temperature THMl °C	°C	Solar	No	No	-
Td: Discharge Gas temp °C	°C	Solar	No	No	-
Te: Evaporation temp °C	°C	Solar	No	No	-
H4: Inverter Operation frequency	Hz	Solar	No	No	-
P1: Compressor running current	A	Solar	No	No	-
Defrosting	-	Solar	No	No	-
Water pump speed	%	Solar	No	No	-
Control Heat. OTC Circuit 1.	-	Solar	No	No	-
Control Circuit 1: Eco mode	-	Solar	No	No	-

Control Circuit 1: Heat ECO Offset Temperature	-	Solar	No	No	-
Control Block menu *6	-	Solar	No	No	-
Control BMS Alarm *4	-	Solar	No	No	-
System Configuration	-	Solar	No	No	-
Software PCB Printed Circuit Board	-	Solar	No	No	-
Software LCD Liquid Crystal Display	-	Solar	No	No	-
Water outlet hp T° Water outlet hp outlet unit temperature	°C	Solar	No	No	-
Ta2: Outdoor Unit Ambient Average Temp.	°C	Solar	No	No	-
O3: Water outlet Temp. 3 Two3 Linked to Inertia Tank	°C	Solar	No	No	-
EVI: Indoor Expansion valve opening	%	Solar	No	No	-
EVO: Outdoor Expansion valve	%	Solar	No	No	-
DI: Cause of stoppage	-	Solar	No	No	-
Water flow level	-	Solar	No	No	-
System status 2	-	Solar	No	No	-
Alarm number	-	Solar	No	No	-
R134a Discharge Temperature	°C	Solar	No	No	-
R134a Suction temperature	°C	Solar	No	No	-
R134a Discharge Pressure	Mpa	Solar	No	No	-
R134a Suction pressure	MPa	Solar	No	No	-
R134a Compressor frequency	Hz	Solar	No	No	-
R134a Indoor Expansion valve 2 opening	%	Solar	No	No	-
R134a Compressor current value	A	Solar	No	No	-
R134a Retry Code	-	Solar	No	No	-
Error: HC cooling below flow minimum temperature	-	Solar	No	No	Sopron, Thessaloniki, USC
Error: Sensor line broken	-	Solar	No	No	Sopron, Thessaloniki, USC
Error: Sensor line short-circuited	-	Solar	No	No	Sopron, Thessaloniki, USC
Error: Overpressure	-	Solar	No	No	Sopron, Thessaloniki, USC
Error: Low pressure	-	Solar	No	No	Sopron, Thessaloniki, USC
Warning: ΔT too high	-	Solar	No	No	Sopron, Thessaloniki, USC
Warning: Night circulation	-	Solar	No	No	Sopron, Thessaloniki, USC

Temperature sensor 1 °C	°C	Solar	No	No	Sopron, Thessaloniki, USC
Temperature sensor 2 °C	°C	Solar	No	No	Sopron, Thessaloniki, USC
Temperature sensor 3 °C	°C	Solar	No	No	Sopron, Thessaloniki, USC
Temperature sensor 4 °C	°C	Solar	No	No	Sopron, Thessaloniki, USC
Temperature sensor 5 °C	°C	Solar	No	No	Sopron, Thessaloniki, USC
Temperature sensor 6 °C	°C	Solar	No	No	Sopron, Thessaloniki, USC
Temperature sensor 7 °C	°C	Solar	No	No	Sopron, Thessaloniki, USC
Temperature sensor 8 °C	°C	Solar	No	No	Sopron, Thessaloniki, USC
Temperature sensor 9 °C	°C	Solar	No	No	Sopron, Thessaloniki, USC
Temperature sensor 10 °C	°C	Solar	No	No	Sopron, Thessaloniki, USC
Temperature sensor 11 °C	°C	Solar	No	No	Sopron, Thessaloniki, USC
Temperature sensor 12 °C	°C	Solar	No	No	Sopron, Thessaloniki, USC
Flow rate sensor 13	l/h	Solar	No	No	Sopron, Thessaloniki, USC
Flow rate sensor 14	l/h	Solar	No	No	Sopron, Thessaloniki, USC
Pressure sensor 19	bar	Solar	No	No	Sopron, Thessaloniki, USC
Pump speed relay 1	%	Solar	No	No	Sopron, Thessaloniki, USC
Pump speed relay 2	%	Solar	No	No	Sopron, Thessaloniki, USC
Output A	%	Solar	No	No	Sopron, Thessaloniki, USC

Counter reading 1	-	Solar	No	No	Sopron, Thessaloniki, USC
Counter reading 2	-	Solar	No	No	Sopron, Thessaloniki, USC
Inverter PV1 voltage	V	Solar	No	No	-
Inverter PV1 current	A	Solar	No	No	-
Inverter PV2 voltage	V	Solar	No	No	-
Inverter PV2 current	A	Solar	No	No	-
Inverter Phase A voltage	V	Solar	No	No	-
Inverter Phase B voltage	V	Solar	No	No	-
Inverter Phase C voltage	V	Solar	No	No	-
Inverter Power grid current	A	Solar	No	No	-
Inverter Phase B current	A	Solar	No	No	-
Inverter Phase C current	A	Solar	No	No	-
Inverter Active power	W	Solar	No	No	-
Inverter Reactive power	W	Solar	No	No	-
Inverter Power factor	-	Solar	No	No	-
Inverter Internal temperature	°C	Solar	No	No	-

Table 8 Variables of MiniStor Energy System

Name	Unit	Cumulative	Aggregate	Site Availability
Operation Mode	-	No	No	Thessaloniki
Swing	-	No	No	Thessaloniki
Accumulated Power	W	Yes	No	Thessaloniki
Set Temperature	°C	No	No	Thessaloniki
User Control	-	No	No	Thessaloniki
Fan Speed	-	No	No	Thessaloniki
Actual Temperature	°C	No	No	Thessaloniki
Status	-	No	No	Thessaloniki

Table 9 Variables of Fan Coil Meter - Thessaloniki Demo Site

4 IoT User Interface Design & Implementation

The purpose of this chapter is to provide an outlook of the various user interfaces developed for interacting with the MiniStor IoT Platform and its subcomponents. The Internet of Things (IoT)-platform allows for user interaction with the system for control and performance monitoring. The IoT infrastructure is composed of a network of devices, and the platform acts as a communication vessel for all system components. To improve the quality of life and user experience, the platform needs to provide a user interface where users can fine tune the usage of the system for optimal operation.

4.1 Functionalities and Graphical layout of the interface

The development of the IoT platform is based on mock-ups defined in the scope of D7.1. The mock-ups were designed so that the flow of the graphical layout promotes logical and intuitive operation of the system. Functionalities of the platform have also been defined in D7.1, by gathering user requirements and usage cases from various users with different backgrounds (the “user stories”). The IoT platform has six main tabs that implement the core functionalities which are detailed in the following sub-sections. The “MiniStor Pilot” tab allows for monitoring of the current status of the system. The “Data Analysis” tab allows for monitoring real time and historical data. “Control Panel” allows the user to control MiniStor’s devices and create operation schedules as well as to enable the automatic system optimization mode. The “Prediction” tab allows to follow a short-term forecast (1 day ahead), real time and historical energy generation and consumption, along with PV Electrical, Thermal power and Irradiance predictions as well as the error deviation of the predicted values from the actual. The “DR Events” tab compiles all the demand-response events that are active or they are about or that have already occurred. Finally, the “MiniStor System” tab presents all metrics coming from the MiniStor system installed at each Demonstration Site, including all PVT, PCM and TCM components. In the following sections, the progress and results of the development of the actual platform are presented.

4.1.1 Login Screen

The platform requires a user authentication process to allow secure access to their data. For the authentication, secure encryption algorithms have been used. Except for logging-in, users can also enable the function to save their credentials to facilitate ease of use for future usage. It can also serve for the event to change their password securely in case they forgot it via the “Forgot Password?” choice.

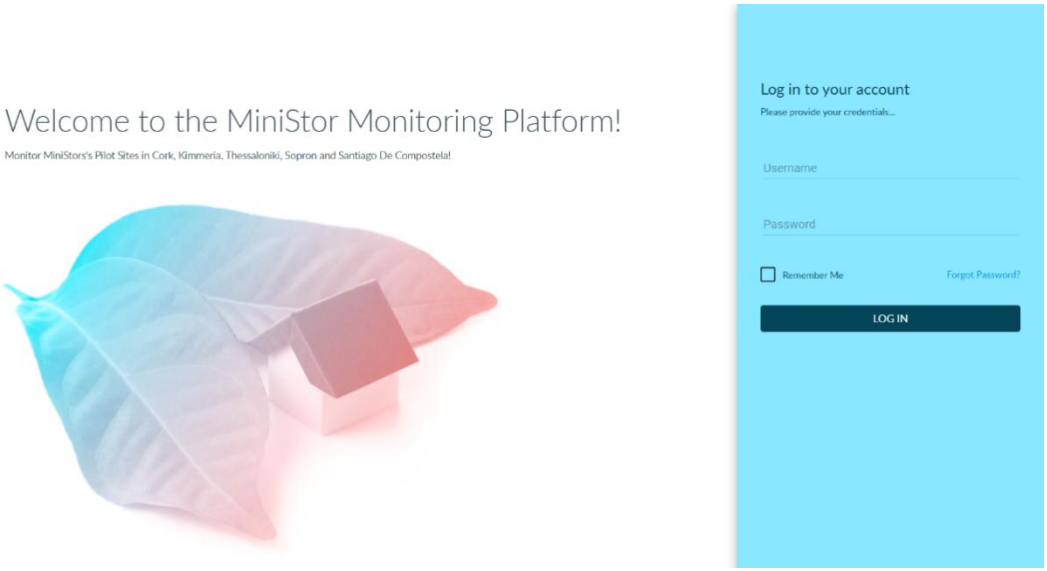


Figure 17 User Interface of IoT platform, Login Page

4.1.2 MiniStor Pilot

After the log-in process, a user can see a summary of the system's and environment's conditions, including both internal and external conditions. External conditions are based on information from a weather station. On this tab, the user can monitor the dwelling's energy consumption. They can also find information about the status of the battery, its stored energy level and whether it is charging or giving energy to the building. Finally, they can see what part of the energy needs of the dwelling are covered by the MiniStor system as well as the amount of energy that is exported or imported from the grid on a daily basis.

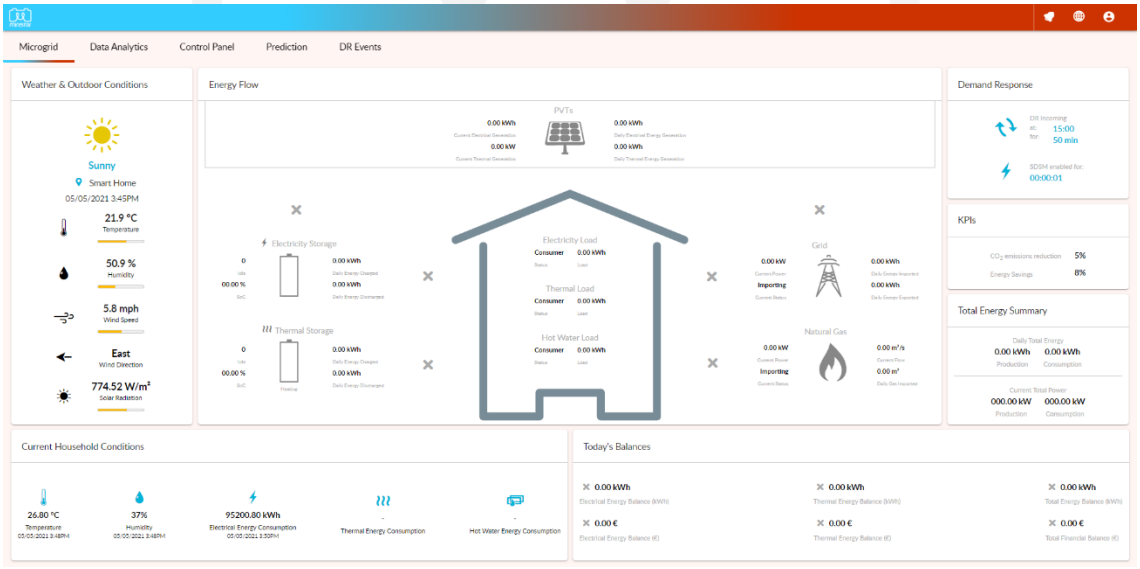


Figure 18 User Interface of IoT platform, Microgrid page

4.1.3 Data Analytics

In this tab, the user can see the electrical and thermal production and consumption of the PVTs as well as the correlated costs from the system's usage in the form of graphs. Also, the user can see the battery status, including the generation, load and current storage. The "Data Analytics" tab allows the user to select between "Real Time" and "Historical Data" modes. For the "Real Time" tab, the user can see the current status of the system, while for the "Historical Data" option, the user can see measurements of any past date by selecting the proper date range.

Figure 19 User Interface of IoT platform, Data Analytics page – Real time

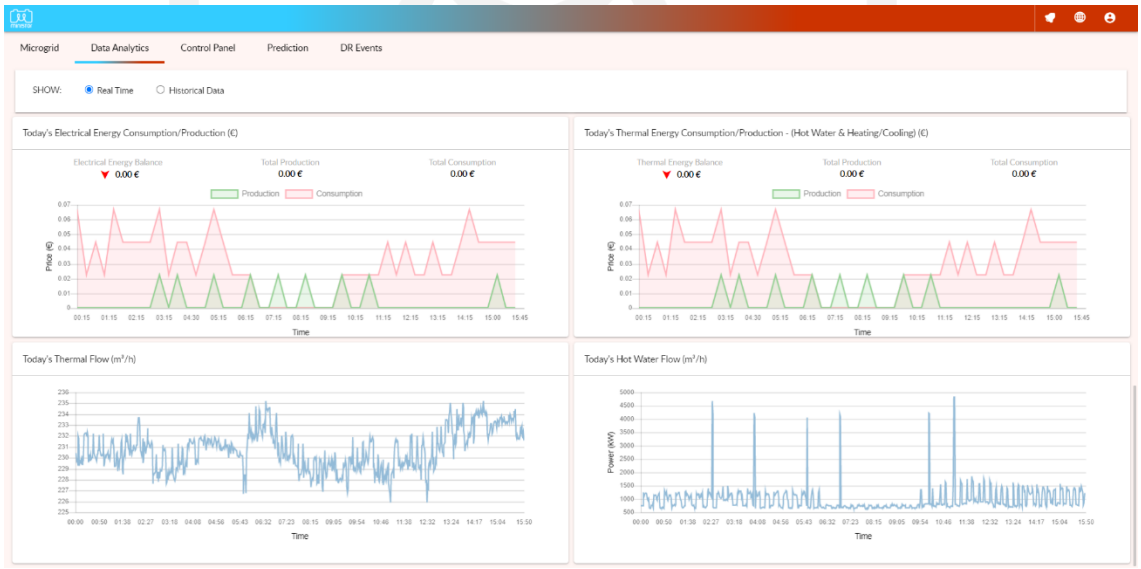


Figure 20 User Interface of IoT platform, Data Analytics page – Additional widgets – Real time

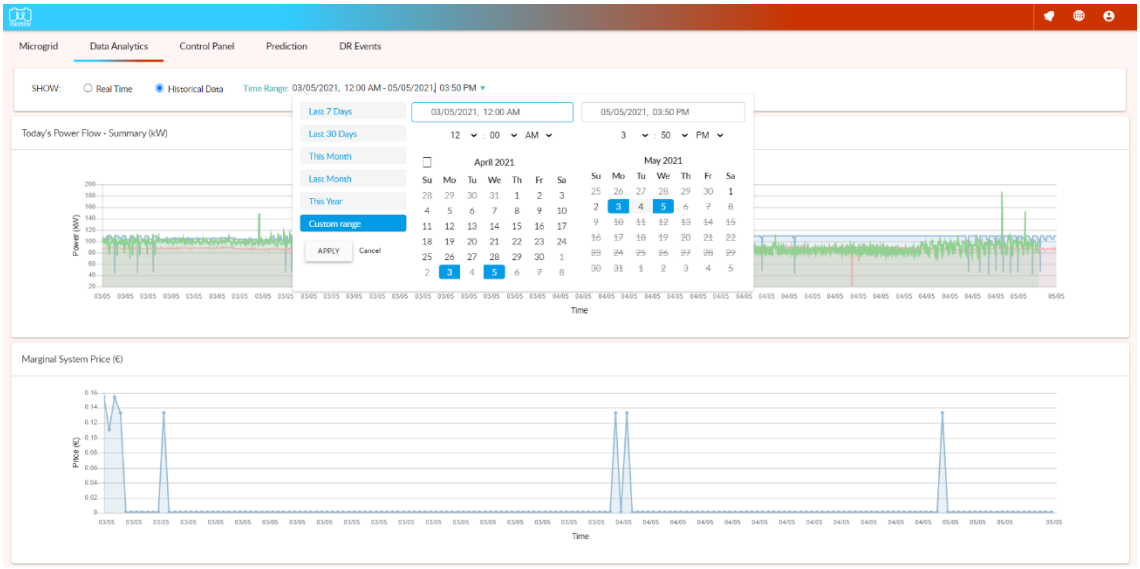


Figure 21 User Interface of IoT platform, Data Analytics page - Historical data

4.1.4 Prediction

In the Prediction tab, the user can see plots that include forecasts for the production and consumption of the total power of the corresponding demonstration site connected with the MiniStor system. The user can see the predicted values for the next day in the “One day ahead” tab. If they want to compare the forecasted values with the real ones, they can select the “Real Time” option. Finally, if the user wants to check the accuracy of the forecasted values compared to the real ones in any specific date, they can choose “Historical Data” and set their desired time range. Note that in both “Real Time” and “Historical Data” modes, the user can see the error deviation of the predictions compared to the real values in a plot.

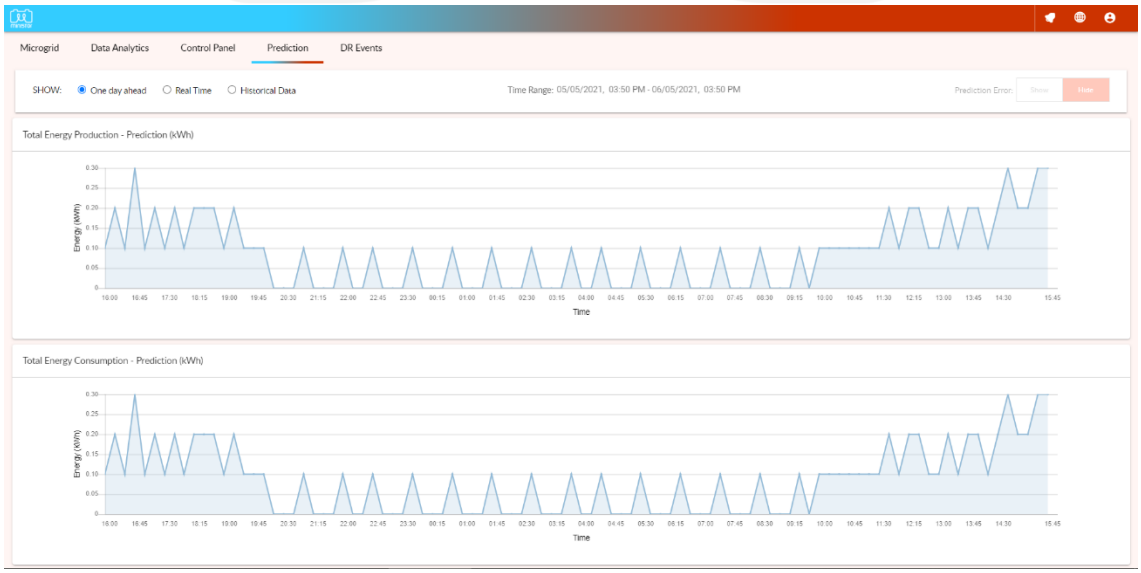


Figure 22 User Interface of IoT platform, Prediction's page - One Day Ahead

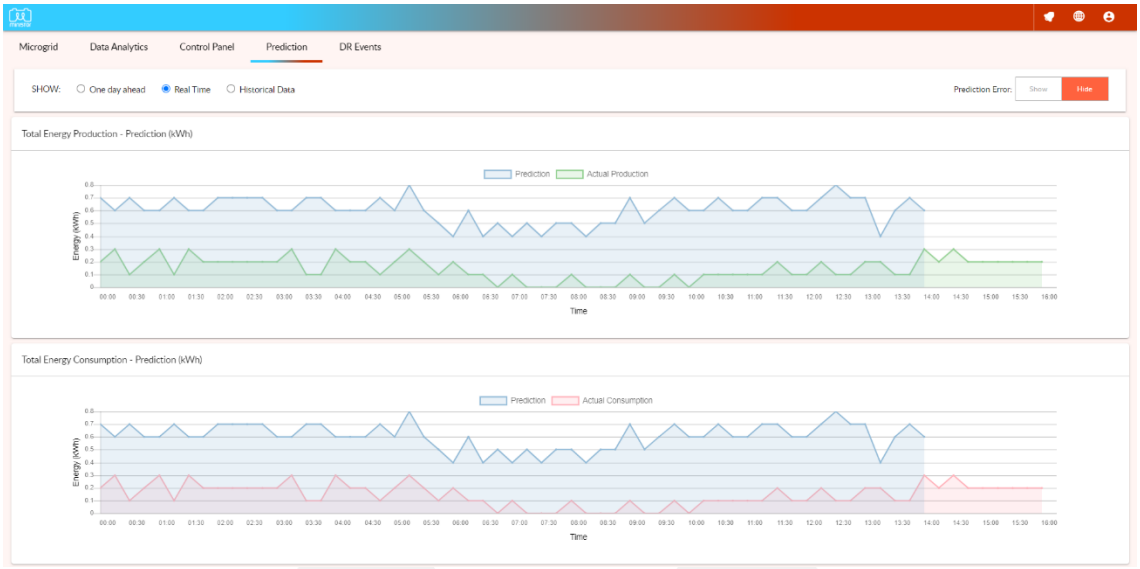


Figure 23 User Interface of IoT platform, Prediction's page - Real Time

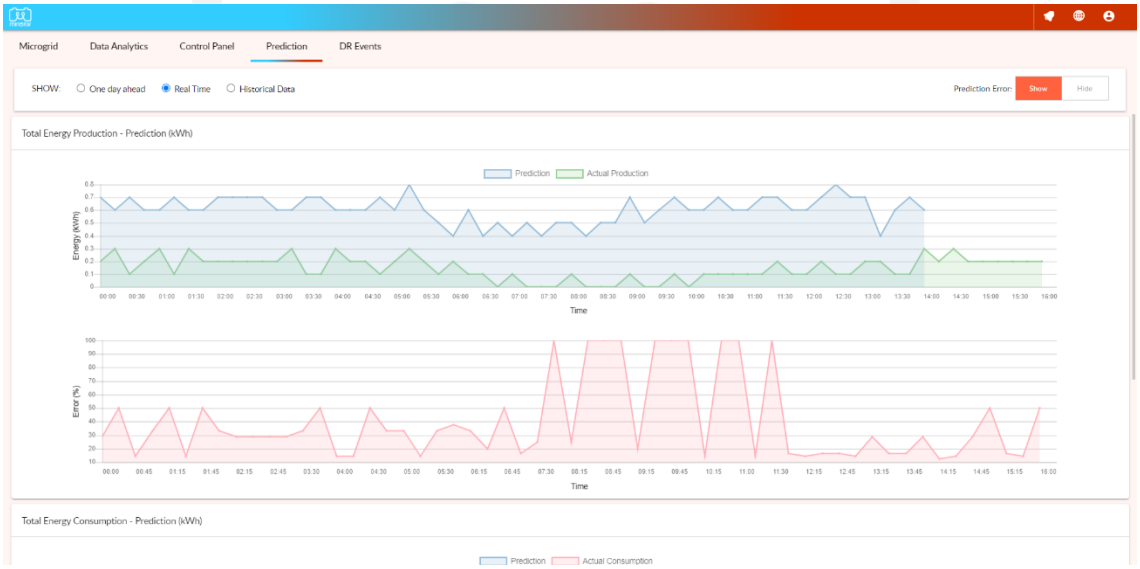


Figure 24 User Interface of IoT platform, Predictions page - Error Deviation Plot - Real Time

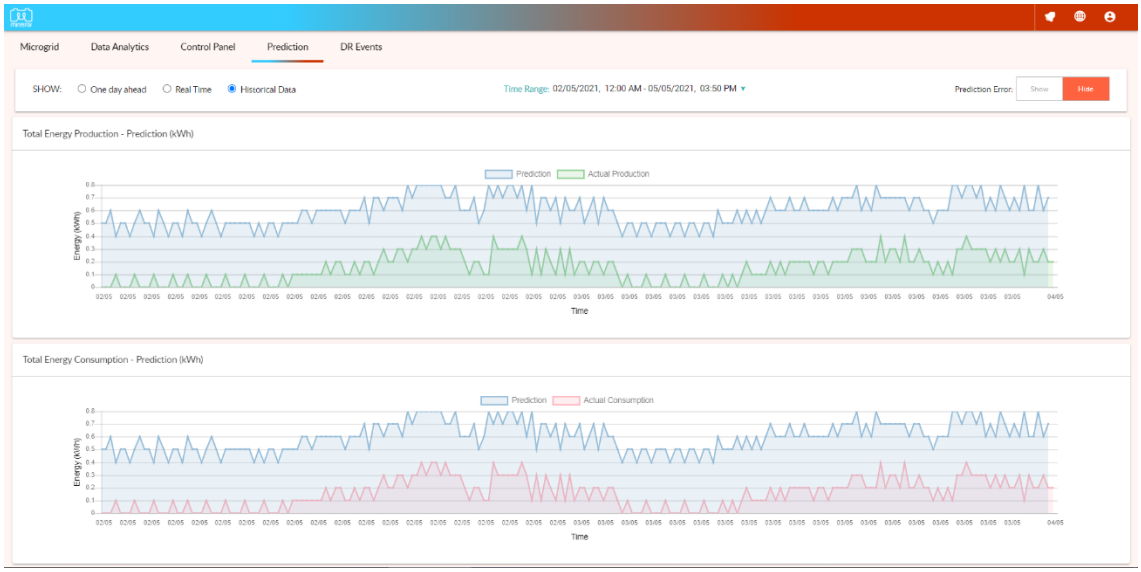


Figure 25 User Interface of IoT platform, Prediction's page - Historical Data

4.1.5 MiniStor System

In the MiniStor System tab, users can view plots representing various data metrics collected from the broad functionalities of the MiniStor System at each corresponding demonstration site. These metrics can be displayed in real time, showing data for the current day, or as historical data, with a user-defined date range.

Below the date selection toolbar, multiple MiniStor system variables are organized into five segments – provided data is available for each segment- based on their respective data sources. The five data categories are: MiniStor Container, Electric Battery, Solar Thermal System, Solar Controller, and PVs. Within each category, plots are further divided into sub-categories according to the nature of the metrics. For example, the plot titled Phase Real Power Consumption shows the real power values for all three phases as well as total real power. Other parameters such as phase currents and frequencies are displayed in separate dedicated plots. A wide range of sub-categories have been selected by default based on their relevance and importance. However, users have the flexibility to add or remove specific plots as needed.

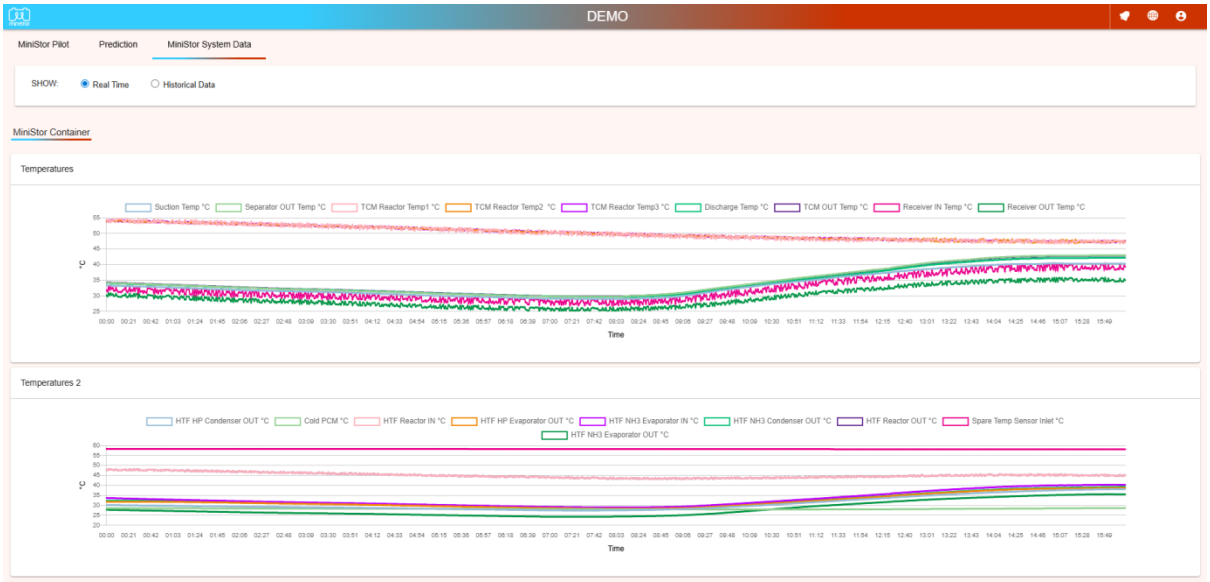


Figure 26 User Interface of IoT platform, MiniStor System page - Real Time data, Ministor Container segment

Figure 27User Interface of IoT platform, MiniStor System page - Historical data, Custom range, Solar Controller segment

5 Components Integrated with the IoT platform

In the following sections, we provide detailed information about the installation status of relevant hardware and software components as well as the development tools that were necessary to provide real-time monitoring data for the IoT platform.

As part of the effort to demonstrate the capability of MiniStor to be accommodated in existing environments as well as in newly constructed buildings, there were considerations for existing IT infrastructure and any environmental monitoring capability at each demo site. In coordination with demo site representative partners, it was established that in two of the demonstration cases the existing building management system can be used to receive all additional measuring devices needed. In the same sites, the system can also be connected to the IoT platform directly. In order to outline why certain development activities were necessary in some cases but not in others, we summarize relevant pre-existing and new infrastructure in the following table.

Demo site	New building	Existing local monitoring system	Local gateway
Cork	No	No	New RaspberryPi installed
Kimmeria	No	Yes	No, direct connection
Santiago de Compostela	No	Yes	No, direct connection
Sopron	Yes	No	New RaspberryPi installed
Thessaloniki	No	Yes	RaspberryPi already present, with data transfer software

Table 10 Summary of pre-existing and new monitoring systems at the demo sites

In the following sections, we report the status of any software components that were required to be developed for integration purposes.

5.1 Use case scenarios, assets and relevant software

In this subsection, we describe the main use cases for each pilot site, the supported hardware assets and the integration status of the hardware components with the platform in regard to their corresponding software components.

The pilot site in Thessaloniki constitutes a pre-demonstration platform mimicking residential usage with main use as offices. For the case of Sopron, the dedicated space that will be served by MiniStor is a residential area. For the case of University of Santiago de Compostela, the pilot site is a university apartment that is going to be used in a family setting. The demonstration site in Cork is a residential house. Finally, for the case of Kimmeria, the dedicated space is dormitory student apartments inside the campus of DUTH.

The demonstration sites have installed sensors to monitor indoor and outdoor conditions. They include electrical energy meters, room sensors for measuring temperature and humidity, environmental sensors to monitor weather conditions, gas and air flow meters (only for Cork and Sopron) and the respective data-logging hardware to save the sensor measurements. The MiniStor system electrical storage (PVTs, battery etc.) was installed in four pilot sites and is also integrated with the platform. More details for the hardware equipment are described in D6.1 [7] and WP3.

The platform is able to support all pilot sites with their respective assets. Also, the platform is built to fulfill identified stakeholders' requirements and the possible use cases as defined in D.7.1, Thessaloniki's and Sopron's pilot sites have already been integrated with their existing assets, while the rest of the pilot sites are also supported. The assets have been defined for each demonstration site, and they also have been mapped to relevant endpoints on the API.

5.2 Local RaspberryPi (RPi) Gateway development

At two of the demo sites, in Cork and Sopron, there was no existing environmental monitoring or smart building management systems installed previous to the project start. This condition required adding a local gateway device in order to provide facilities for the following functions:

- Data readout for measurement devices without direct IP interface (e.g. RS-485)
- Short-term data storage in case of temporary Internet connectivity issues
- Local data backups in order to provide additional long-term data security
- On-site conversion to JSON-format for use with the REST API described in section 3.3
- Providing secure VPN connection for remote configuration

In order to fulfil the above-mentioned roles, it was decided that a small form-factor industrial computer was required, for which the RaspberryPi 3 B+ SBC (Single Board Computer) was selected.

The following table describes the software-related activities for the local RPi.

Activity/Module	Status	Scripting language
Configuration of Elvaco CMe3100 M-Bus data logger	Complete	
Readout of CMe3100 through REST API	Complete	BASH
Readout of DeltaOhm weather station through Modbus RTU	Complete	BASH
Readout of Airvent HVAC unit through Modbus TCP/IP	Complete	BASH
Readout of Davis weather station through Modbus TCP/IP	Complete	BASH
CSV-to-JSON and XML-to-JSON conversion	Complete	BASH
JSON file upload trough REST API	Complete	BASH
CERTH's and DUTH's monitoring censor data upload trough REST API	Complete	Python
Configuration of TMPFS (Temporary File System)	Complete	BASH

Table 11 Status and scripting language of different activities and modules for the local RPi

5.3 Backups

In case of those demo sites where the local RPi gateway is used, there are a couple of implicit data backup mechanisms in operation:

- The CMe3100 data logger automatically keeps every measurement until its storage medium is full, after this it operates in 'First-In-First-Out' mode, providing a rolling window and keeping only

the recent data. Window width for example in the case of Sopron demo site is estimated to be around 30 days.

- On the RPi, the device readout scripts produce CSV or XML files which are kept even after conversion and upload.

There were additional development activities performed in relation to the extension of these measures for improved data redundancy. It is worth mentioning that applicable data protection guidelines were strictly followed when using procedures of such type.

Activity/Module	Automatic download of central IoT data for back-up
Status	Complete
Scripting language	BASH
Description	In order to provide a third location for safekeeping data from possibly other demo sites as well, central data storage will be periodically accessed through a secure channel (REST API) to back-up the data.
Applicable demo sites	All

Table 12 Automatic download of central IoT data for back-up status

5.4 Alerts

As the environmental monitoring period was planned to start before development of the IoT interface could progress to a stage where functions as reports and alerts could be finished, requirements arose to provide a low-level alert system. The goal of this function is to provide means for demo site representatives to be informed as soon as possible and at the development stage, of any hardware or software component malfunction, or prevention of data recording. The following software development actions have been executed:

Activity/Module	Status	Scripting language	Description	Applicable demo sites
Low-level alert module for RPi Gateway	Complete	BASH	Periodic check on the RPi of such conditions as e.g., there are adequate free storage space, VPN connection is active, and central IoT server is accessible. Sends email message if a persistent problem is encountered. This will only work if Internet access of the RPi itself is not prevented.	Cork, Sopron
Mid-level alert module for checking if data uploads are working	Complete	BASH	From a third location, the script checks periodically though the REST API if there are recent data uploaded to the IoT platform for every measurement device in relation of a certain demo site. If some 'latest values' are too old, an email alert is generated.	All

Table 13 Activity/Module for low- and mid-level alert module for RPi Gateway status

6 Collaboration tools for software development

To support a swift development process, toolkits were required for the management of program source files, organization of development activities / sprints and tracking of potentially identified problems and bugs.

In this project, two platforms have been selected due to previous experiences of the development team as well as the assessment of the different options in Error! Reference source not found.:

- *Gitlab*[3]
  Gitlab is an open source, web-based Tool that supports development operations (DevOps) during the lifecycle of a project. As the name suggests, it is based on the git technology and features also wiki-functionalities, issue tracking, continuous integration and deployment pipeline features that support the development team in the integration but also testing and deployment of the developed tools.
- *Openproject*[4]
  To ensure an on-time completion of the tasks as well as facilitate the continuous overview over the ongoing activities, the tool Openproject has been chosen. This tool offers all required functionalities for management of the (development) project.

Name	Description	Pro	Cons
CVS	Well established framework for source code organization and development tracking https://savannah.nongnu.org/projects/cvs	Successfully applied in industry for decades	Last official release in 2008
Git	One of the industry standards for code organization and management https://git-scm.com/	One of the most used software	Bug tracking and wiki functionality not included in tool.
Github	Online platform for source organization and management based on the git technology. Offers also the tracking of issues as well as bugs. https://github.com/	Wide user basis, online solution, worldwide access	Not open-source, local hosting of servers not possible
Gitlab	Open-source tool for the organization of code and issues based on the git technology. https://about.gitlab.com/	Wide user basis, online solution, local hosting possible, open-source tool	Project management tasks (time plans) challenging to implement
OpenProject	Open-source project management software for the organizational aspects of a project. openproject.org	Simple to use, offers all required functionality, can be locally installed	
Monday.com	Monday.com	Intuitive user interface, very wide user basis	Online only solution, no local hosting possible

Table 14 Overview over common source organization and development organization tools.

[3] https://about.gitlab.com/
[4] https://www.openproject.org/

7 Testing Methodology

This section covers the testing methodologies outcomes that have been developed for the integration and acceptance assessments of the system's hardware and software components.

During the project we are following the terminology and guidelines laid down by ISTQB®, the International Software Testing Qualifications Board[5]. One of the key activities in testing management is to create a Test Plan. Its goals are the following:

- Establishing the structure for all the testing efforts related to the development of software modules in the project
- Identifying the scope, i.e. all involved subsystems of the software
- Introducing the testing policy, including approach and strategy
- Presenting a layout for the tests, including:
    - Activities
    - Test types and techniques
    - Roles and responsibilities
    - Schedule

The Test Plan document was maintained continuously throughout the course of development. It had an ever-evolving nature following the progress of testing activities. It also had to address changes in conditions and requirements. First version of the Test Plan for MiniStor project was created and approved in Nov 2020 (M13). Throughout this chapter, we present the plan with summary remarks of the executed testing activities.

7.1 Integration Tests

7.1.1 Schedule

The initial scheduling for integration tests was aimed to conduct at least one cycle of test execution and feedback for most of the software modules before the expected delivery and installation of the main MiniStor equipment.

The Thessaloniki Smart Home test bed was the first one used for the development of the platform as part of the monitoring system that was already installed. In the following section, the results of the integration tests are included regarding each module.

We must note that for some modules more than one development/test/feedback cycle has been applied, when deemed necessary, which does not necessarily imply the quality of the software module is low. Through a more agile coordination between partners, this has been improved in order to be able to test at intermediate stages.

7.1.2 Status of Analysis Phase

Module	RPi Gateway (WOODSPRING)
Status	Analysis completed
Tester	WOODSPRING
Interface(s)	• Measurement/Data logging devices • IoT REST API

[5] https://www.istqb.org/

Approach/Strategy	Methodical search for common or likely types of failures, behavior on unexpected conditions. Examples include malformed input data, connection problems, fault tolerance to other hardware anomalies.
Entry criteria	Module is ready for operation
Exit criteria	Complete
Results	Most serious issue that has been found was vulnerability to storage overflow on the RPi when subjected to long periods without active network connection to the REST service. The problem could be addressed by introducing scheduled file archival strategies. Otherwise, fault tolerance of the system was found to be adequate, with the probability of data loss being very low when coupled with all mentioned backup and alerting measures.

Table 15 RPi Gateway Status

Module	IoT-HEMS Services (CERTH)
Status	Analysis completed
Tester	WOODSPRING
Interface(s)	• WEB interface for user interaction • IoT REST API to connect with other services •
Approach/Strategy	• WEB: Search for common or likely types of failures, also exploratory testing, especially in relation to user requirements (E.g. analyzing whether anything could go wrong if a user makes a mistake in pre-defined usage scenarios) • REST API: Methodical search for cases where input data fails to be recorded or displayed correctly in the user interface • Supplementary: testing of security features
Entry criteria	Module is ready
Exit criteria	Complete
Results	The WEB interface was found to perform satisfactorily. Minor improvements were suggested only in relation to browser compatibility (mobile devices) and accessibility (scaling). The REST API showed good resistance to anomalous input, and all data plots displayed were consistent when compared to raw backup series.

Table 16 IoT-HEMS Services Status

Module	1st Level Control (CARTIF)
Status	Analysis completed
Tester	CARTIF, WOODSPRING
Interface(s)	• Low-level interfaces with sensors/actuators/controllers of PCM, TCM, PVT • Connection to HEMS
Approach/Strategy	Methodical search (malformed input, connection problems), testing for compliance to project-specified standards regarding possible MiniStor states and operation modes.
Entry criteria	• Configuration/program needs to be ready to be installed on PLC device • HEMS interface needs to be completed for manual mode • Otherwise, ready for Design phase
Exit criteria	Complete

Results	As this module was identified as one of the most important from operational safety standpoint, it was subjected to multiple phases of rigorous testing. CARTIF performed black-box validation on the local control strategies, with special attention to the auxiliary PLC (safety backup mechanism). WOODSPRING assisted in communication tests between the controller and local monitoring systems in the early phases of development. Later they carried out static analysis on the Node-RED control logic and verified its adherence to required code quality standards, as well as it being fully consistent with the theoretical control schemes that have been laid down and agreed upon earlier (see D5.1 "Initial design of the Energy Management System").

Table 17 1st Level Control Status

Module	2nd Level Control (CARTIF)
Status	Analysis completed
Tester	WOODSPRING
Interface(s)	• HEMS platform • 1st Level Control
Approach/Strategy	Testing tolerance for anomalous data, checking for compliance to project-specified standards regarding possible control states and operation modes.
Entry criteria	• Preliminary implementation required to be able to start • Otherwise, tests could be conducted incrementally
Exit criteria	Complete
Results	Interactions with the high-level control system have been checked by the way of static analysis of both the end-user facing HEMS IoT-platform code and relevant input nodes in the low-level PLC control flow, to be able to confirm that malformed data or lack of input cannot put these systems into an invalid state.

Table 18 2nd Level Control Status

Module	KPI Calculations (HSLU)
Status	Analysis completed
Tester	WOODSPRING
Interface(s)	HEMS platform
Approach/Strategy	Analytical testing of correct implementation of project-specified KPIs, also checking tolerance for anomalous data.
Entry criteria	The list of KPIs that are to be calculated continuously
Exit criteria	Complete
Results	Both static analysis and unit tests have shown that the implementation of KPI calculations that are visualized in the platform user interface are indeed in agreement with specified formulas laid down as part of WP6 (see D6.1 "Design of the monitoring system and KPI definition").

Table 19 KPI Calculations Status

Module	TCM, PCM, PVT (CNRS, SOFRIGAM, PSYCTOTHERM, ENDEF)
Status	Analysis completed
Tester	WOODSPRING and relevant technology providers
Interface(s)	1st Level Control PLC

Approach/Strategy	Testing tolerance and endurance to project specified standards, regarding possible control states and operation modes.
Entry criteria	• Safety standards adherence • Module ready for operation
Exit criteria	Complete
Results	As these subsystems use additional low-level control devices (that are in direct contact with the 1st Level Control PLC), relevant partners had to make sure these components comply to standards that have been identified as part of WP2 (see D2.3 “Analysis of relevant legislation and standards for system operation”). Static analysis on the 1st Level PLC code included thorough checking of nodes connected to these inputs, in order to exclude the possibility of the system entering an invalid state.

Table 20 PCM, TCM, PVT Status

7.2 Acceptance Tests

7.2.1 Activities

Regarding the acceptance tests, WOODSPRING has accepted all the major responsibilities. Similarly to integration tests, progress monitoring and controlling duties have involved CERTH as Task Leader, with participation of CARTIF as Work Package Leader.

Representatives of demo site owner organizations have been kindly asked to take part in the execution phases. Also, other interested colleagues in the project consortium were welcomed to join as a tester.

7.2.2 Entry and Exit Criteria

Execution of the tests started as soon as the UI for the HEMS platform has been accessible remotely, and synchronization of monitoring data was solved in relation to the Sopron demo building (which is managed by WOODSPRING). The latter is required to be able to evaluate the user interface in relation to the actual observed conditions.

During the early analysis stage partial coverage from data collection has been considered i.e., before all the associated modules are ready.

A set of appropriate exit criteria was set up for the tests. The final goal was to improve the system until all participant testers rate the package favorably in every way, but given that at least some of the rating system will inherently be subjective, good threshold values must be devised for the stop condition.

7.2.3 Test Techniques

Both User Acceptance Testing and Operational Acceptance Testing were aimed to be conducted. In the case of WOODSPRING, they also acted as system operators of the Sopron demo site. Both functional and non-functional attributes in the tests have been included. Tests were executed manually by persons with different backgrounds and thermal energy perspectives.

Tests have been based on user stories that were documented in Work Package 7 of the project. Test reports have been recorded as questionnaires with response types such as checklists, scorecards with "1-to-10" rating type items, Likert scale and free text.

As part of the Operators Acceptance Testing, such responses have been investigated in the analysis phase.

7.2.4 Status of Analysis

In this section we present our approach in relation to the analysis phase of acceptance tests.

With regard to already implemented functionality, the following questions have been proposed to be included in the final round of acceptance tests.

	User Type	Epic	User Story
1.	Resident/Building Manager	Monitoring	As a user, can I view the temperature of each room in my apartment?
2.	Resident/Building Manager	Monitoring	As a user, can I see the temperature or humidity data in the past? E.g. in a given day, so that I can monitor the conditions of my building.
3.	Resident/Building Manager	Monitoring	As a user, can I see the energy consumption of the building, so that I can monitor its behavior?
4.	Resident/Building Manager	Monitoring/Data Analytics	As a user, can I see the monitoring data in the form of graphs so that it is easier to observe the state in a certain period?
5.	Resident/Building Manager	Monitoring/Data Analytics	As a user, can I choose the period for which I would like to see the data?
6.	Resident/Building Manager	Monitoring	As a user, can I see the status of my building's conditions in a summary, so that I can easily grasp the overall picture of its condition?

Table 21 Questions for first round of acceptance tests

The results are included in Deliverable D7.2 "Design evaluation through user validation in demonstration sites", where the work conducted for the user validation is presented and thus, within whose related task activities, the questionnaire has been circulated among the pilot participants.

Conclusions

This document describes the outcome of functionalities added to the initial implementation of the IoT platform and its user interface that were developed within Task 5.3, allowing users to monitor performance and control the system. These functionalities optimize system usage through an intuitive user interface that receives inputs from the monitoring system. The deliverable describes the methodology that was followed to reach the Platform's final architectural and development state. It also includes the functions and actions that the IoT platform offers as well as a description of the data layer. This data layer includes the architecture of the data mechanisms as well as documentation for the API endpoints.

In addition, testing methodologies and outcomes were shown, as defined in Task 5.4. Their aim is to evaluate interfaces and functionality of components and ensure the organized and seamless integration of both hardware and software components. The main pages of the user interface are programmatically completed according to the existing mock-ups. All five pilot sites are integrated with the user interface and the platform is able to seamlessly support all demo sites that have installed sensors with a simple and fast procedure.

References

[1] "MiniStor - Deliverable D7.1 Design evaluation through user validation in pre-demonstration site".
[2] https://www.digite.com/agile/scrum-methodology/
[3] https://www.wrike.com/project-management-guide/faq/what-is-agile-methodology-in-project-management/
[4] "MiniStor - Deliverable D1.7 Data management plan".
[5] https://www.lbmc.com/blog/three-tenets-of-information-security/
[6] Cai, L., & Zhu, Y. (2015). The Challenges of Data Quality and Data Quality Assessment in the Big Data Era. Data Science Journal, 14, 2. DOI: http://doi.org/10.5334/dsj-2015-002 (Available at https://datascience.codata.org/articles/10.5334/dsj-2015-002/)
[7] "MiniStor - Deliverable D6.1 Design of the monitoring system and KPI definition."