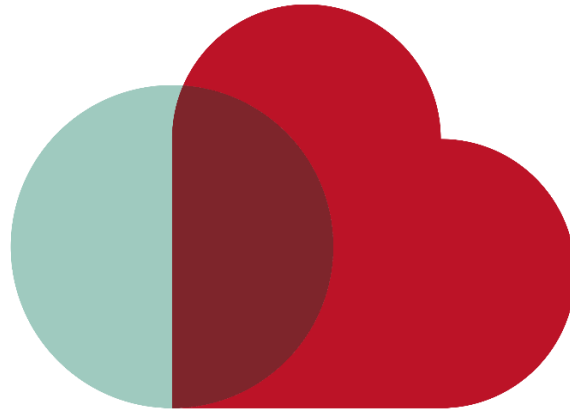




retention

SMART HEART FAILURE MANAGEMENT

Horizon 2020 Project RETENTION

“HEART FAILURE PATIENT MANAGEMENT AND INTERVENTIONS USING CONTINUOUS PATIENT MONITORING OUTSIDE HOSPITALS AND REAL WORLD DATA”

Research and Innovation Action
H2020-SC1-BHC-2018-2020
GA 965343

Duration: 48 months from 01/05/2021

Coordinator: Institute of Communication and Computer Systems

Deliverable ID.:	D6.1	
Deliverable title:	RETENTION Interfacing, Device Federation and Visualisation Components v1	
Planned delivery date:	31/12/2022	
Actual delivery date:	28/02/2023	
Deliverable leader:	AEGIS	
Contributing partners:	AEGIS, DM, FORTH, SIESLR	
Dissemination Level:	X	PU = Public;
		CO = Confidential
		CI = Classified



This project has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 965343.

This deliverable reflects only the authors' view and the Commission is not responsible for any use that may be made of the information it contains.



Document information and history

Deliverable description (from DoA)

Delivery of the first version of the RETENTION Mobile Application, Gateway and smart devices federation, as well as CSB and GIC dashboard and visualisation components.

Version N.	Date	Author [Person and Organisation]	Reviewer [Person and Organisation]	Milestone*	Notes
V. 0.1	20/10/2022	Michalis Vakalellis (AEGIS)		ToC	
V. 0.2	12/12/2022	I. Nicolae, G. Danciu (SIESRL), Athanasios Pardalis, Yorgos Goletsis, Mary Roumpi (FORTH), George Del Toro (DM), Giannis Ziazaris (AEGIS)		1st draft version	1st round of contributions
V. 0.3	15/12/2022	I. Nicolae, G. Danciu (SIESRL), Athanasios Pardalis, Yorgos Goletsis, Mary Roumpi (FORTH), George Del Toro (DM), Giannis Ziazaris (AEGIS)		2nd draft version	2nd round of contributions
V. 0.4	18/12/2022	I. Nicolae, G. Danciu (SIESRL), Athanasios Pardalis, Yorgos Goletsis, Mary Roumpi (FORTH), George Del Toro (DM), Giannis Ziazaris (AEGIS)		3rd draft version	Updated contributions
V. 0.5	27/12/2022		I. Basdekis (STS)	Internal Review by TCo	Comments/Improve ments
V. 0.6	31/01/2023	Michalis Vakalellis (AEGIS), George Del Toro (DM)		Updated version	TCo review comments addressed
V. 0.7	06/02/2023		N. Vassileiou, I. Kouris, M. Haritou (ICCS)	Final Review by PCo	Review comments provided, changes asked
V. 0.8	23/02/2023	Michalis Vakalellis (AEGIS)		Updated version	Review comments addressed
V. 0.9	24/02/2023	Miltiadis Kokkonidis (AEGIS)		Final version	Quality Assurance
V. 1.0	28/02/2023		M. Haritou (ICCS)	Deliverable approved	Deliverable Submitted



Table of Contents

Executive Summary	10
1 About this Document	11
1.1 Scope of deliverable	11
1.2 Relationship to other RETENTION deliverables.....	11
1.3 Structure of the document.....	12
2 RETENTION Edge Mobile Application	13
2.1 Description and objectives	13
2.2 Mobile application tools and libraries.....	14
2.3 User journeys.....	15
2.4 When to use it for each device.....	30
2.5 Data flows.....	31
2.6 Configuration details and APIs	31
3 RETENTION Local Home Gateway	33
3.1 Description and objectives	33
3.2 Data flow	34
3.3 Device federation	37
3.4 Configuration details and APIs	44
4 RETENTION Clinical Site Backend Dashboard & Visualisations	50
4.1 Description and objectives	50
4.2 Dashboard tools and libraries	50
4.3 User journeys.....	51
4.3.1 Clinician's use cases.....	52
4.3.2 Risk assessment generated by AI	62
4.3.3 Technical use cases.....	63
4.4 Data flows.....	68
4.5 Configuration details	70
5 RETENTION Global Insights Cloud Dashboard & Visualisations	71
5.1 Description and objectives	71
5.2 Dashboard tools and libraries	71
5.3 User journeys.....	71
5.4 Data flows.....	73



5.5	Using MLFlow	76
5.6	Using Jupyter Notebook	81
6	Conclusions and next steps	82
7	Appendix.....	83
7.1	Installation instructions for the Local Home Gateway	83
7.2	API data	84



Table of Abbreviations

AI	Artificial Intelligence
API	Application Programming Interface
BLE	Bluetooth Low Energy
CSB	Clinical Side Backend
DoA	Document of Action
DSS	Decision Support System
FHIR	Fast Healthcare Interoperability Resources
FORTH	Foundation for Research and Technology Hellas
GATT	Generic ATtribute
GIC	Global Insights Cloud
GUI	Graphical User Interface
HF	Heart Failure
http(s)	(Secure) Hyper Text Transfer Protocol
ICCS	Institute of Communication & Computer Systems
JSON	JavaScript Object Notation
LVAD	Left Ventricular Assist Device
MAC	Media Access Control
ML	Machine Learning
pkl	Python Pickle File
REST	Representational state transfer
RWD	Real World Data
SD	Secure Digital
SHAP	SHapley Additive exPlanations
SIESRL	Siemens SRL
STS	Sphynx Technology Solutions
URL	Uniform Resource Locator
USB	Universal Serial Bus



Table of Figures

Figure 1: RETENTION architecture	11
Figure 2: RETENTION mobile application login page.....	15
Figure 3: RETENTION mobile application main screen.....	16
Figure 4: RETENTION mobile application menu tab.....	16
Figure 5: Measures option	16
Figure 6: Measures screen	17
Figure 7: Smartwatch button	18
Figure 8: Scan button	18
Figure 9: Available devices to be connected	18
Figure 10: Personal information screen	19
Figure 11: Paired devices button.....	19
Figure 12: Synchronized device	20
Figure 13: Charts button.....	20
Figure 14: Available charts options	21
Figure 15: Heart rate chart example	21
Figure 16: Activity button.....	22
Figure 17: Activity results example	22
Figure 18: Oxygen level button	23
Figure 19: Scan button	23
Figure 20: Oxygen level result example	23
Figure 21: Oxygen level button	24
Figure 22: Measurement history button.....	24
Figure 23: Body weight button.....	24
Figure 24: Scan button	24
Figure 25: Weight measurement example	25
Figure 26: Weight button	25
Figure 27: Measurement history example	26
Figure 28: Blood pressure button.....	26
Figure 29: Scan button	26
Figure 30: Blood pressure meter readings example	27
Figure 31: Blood pressure button.....	27



Figure 32: Blood pressure history measurements example.....	28
Figure 33: Temperature button.....	28
Figure 34: Temperature module main screen.....	29
Figure 35: Menu tab	29
Figure 36: VAD button	29
Figure 37: VAD module main screen	30
Figure 38: Measurement flow	31
Figure 39. The temperature and humidity sensor wirelessly connected to Raspberry Pi.	34
Figure 40: Data flow of the initial setup and configuration of the Local Home Gateway.....	35
Figure 41. Data flow of retrieving data by the Local Home Gateway and posting to the GIC database.	36
Figure 42. Air quality index definition (source: https://openweathermap.org/api).....	38
Figure 43. Terminal output of the incoming sensor's data.	39
Figure 44. Local data storage structure.....	40
Figure 45. The generation of API keys.....	41
Figure 46. Open weather map API calls.	42
Figure 47. Geocoding API responses.	42
Figure 48. API call to retrieve weather data based on longitude and latitude.	42
Figure 49. Response of the API call.	42
Figure 50. API call to retrieve the air pollution data.	43
Figure 51. Response of the API call for the air pollution.....	43
Figure 52. The .json format of external temperature, humidity and pollution index variables.	44
Figure 53. The technician tools for the Local Home Gateway setup.	45
Figure 54. Pseudocode for the sensor data API functionality.....	46
Figure 55. Pseudocode for the configuration API functionality.	46
Figure 56. Pseudocode for the watchdog process.	47
Figure 57. Pseudocode for post sensor data API functionality.	47
Figure 58. Pseudocode for weather data API functionality.	48
Figure 59. Pseudocode for the post weather data API functionality.	48
Figure 60. Pseudocode for the memory handling API.....	49
Figure 61: AVT Technology Stack	51
Figure 62: CSB Frontpage for the Role 'Clinician'	52
Figure 63: Search for specific patient.....	52
Figure 64: Patient creation screen	53



Figure 65: Patient's basic info fields.....	54
Figure 66: Patient monitoring screen.....	55
Figure 67: Patient's baseline (part 1)	56
Figure 68: Patient's baseline (part 2)	57
Figure 69: Visits history	58
Figure 70: New visit creation.....	58
Figure 71: CSB Notification Centre	59
Figure 72: Patient's notification screen.....	60
Figure 73: Patient's events history	61
Figure 74: Creation of a new event for a patient	61
Figure 75: Automatic messages design	62
Figure 76: "Heart Failure" Scenario workflow.....	62
Figure 77: Clinical studies screen.	63
Figure 78: Login and/or Sign Up page of CSB.....	64
Figure 79: Sign Up page for a new user.....	64
Figure 80: End-user management (Admin role).....	65
Figure 81: Grant access after reviewing (Admin role).....	65
Figure 82: Technician's Dashboard	66
Figure 83: Manage end-user account information	67
Figure 84: Patient's assigned devices.....	67
Figure 85: Assign a new device to patient.....	68
Figure 86: Clinician actions regarding patient data.....	69
Figure 87: Clinician Prediction Flow	70
Figure 88: GIC login / sign up screen.....	72
Figure 89: User registration form in GIC	72
Figure 90: GIC data scientist's tools	73
Figure 91: Model management flow	74
Figure 92: Data flow from device to the RETENTION repos.....	75
Figure 93: ML Flow Experiments	76
Figure 94: List of models, ML Flow Models	76
Figure 95: Model versioning details for 'Survival Rate' prediction study	77
Figure 96: Model performance, KPI	78
Figure 97: Model's metrics.....	78



Figure 98: Model artifacts	79
Figure 99: Confusion Matrix	80
Figure 100: SHAP results	80
Figure 101: “Time to death” scenario from a data scientist perspective.....	81



List of Tables

Table 1: Timeframe of measurements	30
--	----



Executive Summary

RETENTION envisions to deliver new disease insights and personalised analytics to Heart Failure (HF) patients exploiting the real-world data gathered from various medical and smart devices. The collection of continuous data from the daily life of patients with HF later on will be analysed to obtain the necessary evidence to provide personalised interventions.

Within this context, the RETENTION platform aims to support the clinical decision by the provision of the following interfacing components:

- A mobile application (for patients and their caregivers) to enable the collection of devices/sensors usage data;
- A local home gateway to be installed in patients' homes for local data aggregation, pre-processing and communication to/from the RETENTION platform;
- A (per clinical site) Clinical Site Backend (CSB) and a (shared) Global Inside Cloud (GIC), connecting all kinds of different measurements, metrics, and other types of data collected, offering Application Programming Interfaces (APIs), and services in support of various workflows foreseen in the RETENTION platform and allowing user interaction via the corresponding dashboards:
 - i) the CSB Dashboard primarily for clinicians but also for technicians supporting patients with the setting up of their devices and
 - ii) the GIC Dashboard supporting Machine Learning (ML) and data analysis experts in the creation of added value from the anonymised data from the various clinical sites and policy makers consulting the GIC visualisations.

The objective of this deliverable is to describe and demonstrate the individual components that are the interfacing components of RETENTION architecture between end users (study participants, clinicians, data analysts, technicians, etc) and the services that RETENTION platform supports. Specifically, it describes the mobile application, the local home gateway and the main dashboards of the Clinical Site Backend and the Global Insights Cloud respectively.

The first version of these components is presented in this deliverable, highlighting the objectives, the user journeys that they serve, the data flow followed, and the configurations details and APIs needed for the successful integration with the rest of the RETENTION components. Specifically, Section 2 deals with the RETENTION mobile application, Section 3 with the local home gateway and Sections 4 & 5 with the visualisations of Clinical Site Backend and the Global Insights Cloud respectively.

1 About this Document

1.1 Scope of deliverable

The current report D6.1 “RETENTION Interfacing, Device Federation and Visualisation Components v1”, elaborates on the development and the current status of the 1st version of the interfacing components been developed (see Figure 1) in the context of WP6 “Enabling Technologies & Secure, Privacy-aware by Design Data Handling” in support of the main Pilots of RETENTION clinical study, and has ties with the following tasks:

- Task 6.1 - RETENTION Edge Mobile Application
- Task 6.2 - RETENTION Edge Gateway & Devices Federation
- Task 6.3 - RETENTION Clinical Site Backend Dashboard & Visualisations
- Task 6.4 - RETENTION Global Insights Cloud Dashboard & Visualisations

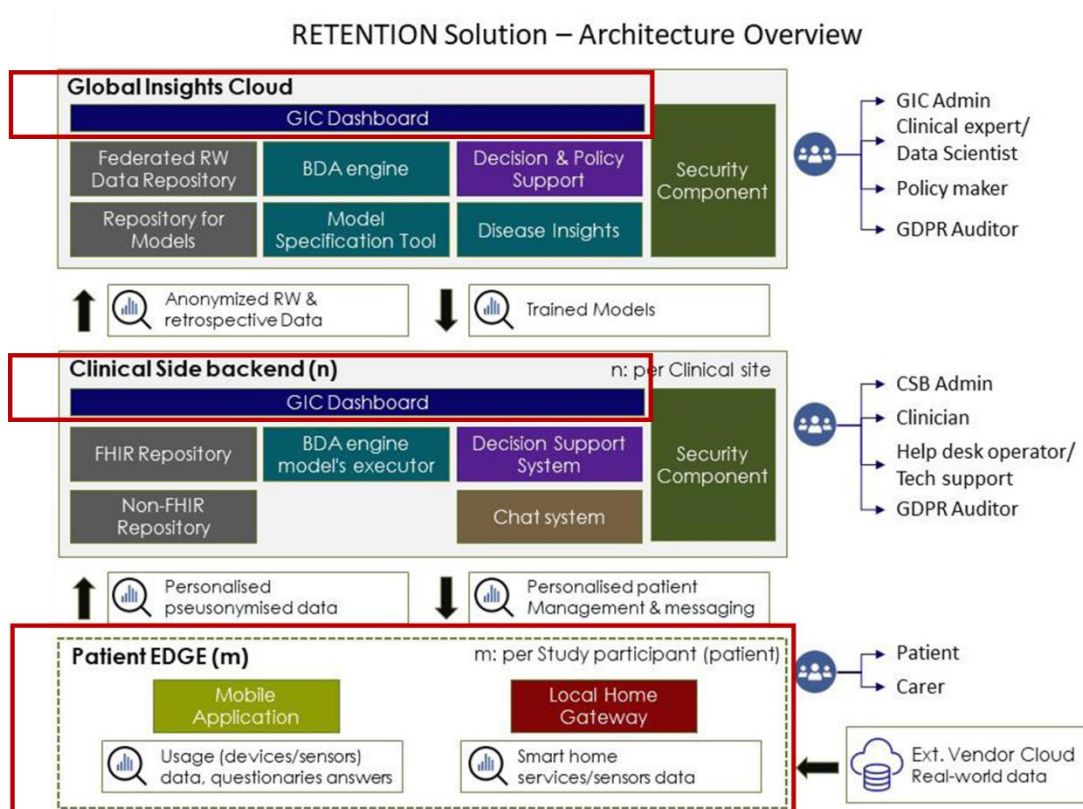


Figure 1: RETENTION architecture

The 2nd and final iteration will be presented within version D6.3 “RETENTION Interfacing, Device Federation and Visualisation Components v2” (due in M36).

1.2 Relationship to other RETENTION deliverables

The work presented in this deliverable is based on the RETENTION architecture, as it is presented in D3.2 “RETENTION Architecture”, and realises the main components that are responsible for interfacing with RETENTION end users.



In addition, this deliverable is related to:

- **D3.2 RETENTION Architecture**, which defines the overall architecture of the RETENTION platform;
- **D4.1 RETENTION Data Model**, which specifies the Data Model for FHIR (Fast Healthcare Interoperability Resources) and non FHIR repos;
- **D4.2 RETENTION Data Management enabling mechanisms version 1**, which describes RETENTION data repositories, the data transfer and handling mechanisms and the FHIR implementation guide that facilitates the data exchange;
- **D5.1 RETENTION Analytics & Decision-making enabling mechanisms version 1**, which describes the delivery of the models, processes and components enabling the analytics, decision-making, and data and model sharing capabilities of RETENTION platform;
- **D6.2 RETENTION Security & Privacy by design enabling mechanisms version 1**, which describes the Security & Privacy mechanisms of RETENTION platform;
- **D7.1 Integrated RETENTION platform version 1** which includes the delivery of the first version of the integrated RETENTION platform involving the first release of all components of developed in WP4, WP5 and WP6.

1.3 Structure of the document

This deliverable is structured in six (6) core sections and an appendix. **Section 1** introduces this document and provides details about the goals of this deliverable, its relation to other project deliverables and summarises the structure of the document. **Section 2** focuses on the RETENTION Mobile Application and provides a summary of its objectives, the user journeys, the data flow, configuration details and APIs used. **Section 3** describes the Raspberry PI-based Local Home Gateway and the processes that run on it (the installation instructions are presented in the appendix). **Section 4** and **Section 5** present the current versions of Clinical Site Backend Dashboard and the Global Insights Cloud Dashboard respectively. The user journeys are accompanied by screenshots and figures taken by the actual demonstration environment of CSB and GIC that are part of the RETENTION virtual Pilot of Pilots. Finally, **Section 6** concludes the document and provides general remarks of the presented work. Finally, the Appendix provides details on the Local Home Gateway installation instructions and the APIs used in the edge mobile application.



2 RETENTION Edge Mobile Application

This section elaborates on the implemented functionality towards the objectives of WP6, and in particular those of T6.1 that are related with the delivering of the RETENTION Edge Mobile Application. This mobile application constitutes the main means of interfacing with all RETENTION study participants (particularly HF patients and – in cases where they cannot use the services themselves – their carers), as specified in the requirements (D3.1) and architecture (D3.2) deliverables.

2.1 Description and objectives

Task 6.1 focuses on designing, developing, and delivering the RETENTION Edge Mobile Application (Mobile app). The Mobile app is meant to feature a multilingual user-friendly interface that gathers and transmits usage data (for previously paired devices/sensors) to CSB (the particular CSB instance with which the specific patient is associated with), inform the end-users (i.e., main Pilots study participants/cares) of the important parameters regarding their health and well-being and the management of their conditions, and deliver personalised interventions.

The main goal of the Mobile App is to interconnect with wearable (bioelectronic) and medical devices to collect their respective data streams, as well as provide supporting services such as information gathering (e.g., questionnaires answers) while the end-user is at home. Considering the non-technical background and the age of most study participants and the fact that the utilisation of (m)IoT devices may present usability challenges for older adults, the design of the interface of the Mobile App will maintain a high level of abstraction and focus on usability, (note: usability testing to validate findings is scheduled to be performed within WP8).

With regards to the data been transmitted to the CSB instance, the Mobile app structures and timestamps data in such a way that can be stored in the (the FHIR or the non-FHIR) CSB repositories and in turn used for ML analytics. In addition, it applies data encryption via a 256-bit cryptographic key stored in a protected format to prevent unauthorised access. The structuring of the data involves their association to a patient identifier (i.e., Patient-Id1), while in some cases (i.e., temperature, pollution sensors' data) micro-aggregations are being performed (defined in D3.1) prior to their transmission to the CSB.

Last but not least, the communication between the Mobile app is implemented using the HTTPS protocol¹ and RESTful web services, which is a way of providing interoperability between computer systems on the Internet.

The Mobile app supports the following functions:

- login (of authorised end-users) to the RETENTION solution via the app;
- collection of usage data and feedback from patients or carers (on behalf of);
- connection via secure channel (i.e., HTTPS) with the CSB;
- visualisation of all data collected/transmitted to the CSB;

¹ HTTPS protocol: <https://en.wikipedia.org/wiki/HTTPS>



- pairing with the devices/sensors supporting the objectives of the RETENTION project (via Bluetooth Low Energy²). In particular:
 - Weight Scale: measures participants body weight;
 - Smartwatch: data of activity (steps, distance, stairs), sleep (hours and quality) and heart rate monitoring;
 - Blood Pressure Monitor: blood pressure and heart rate monitoring;
 - Oximeter: oxygen saturation, heart rate;
 - VAD data for the VAD patients (manual entry by the end-user);
 - Body temperature (manual entry by the user).

2.2 Mobile application tools and libraries

RETENTION Edge Mobile Application uses a series of application development tools and libraries in order to provide a rich set of functionalities on Android smartphones.

A list of the application tools and libraries that are used follows.

Android Studio and Java

Android Studio is an integrated programming environment (IDE) for developing applications on the Android platform. It provides a unified environment where the user can build apps for Android phones, tablets, Android Wear, Android TV and Android Auto. Support for modules allows the user to divide the project into units of functionality that can independently be built, tested and debugged. It is available for Windows, Mac OS X and Linux.

Java is a high-level, class-based, object-oriented programming language, as consequence was for a long time the official language for Android mobile application development. The Java Development Kit (JDK) is free, under an open-source license and available at jdk.java.net.

Android Studio is used for RETENTION's Edge Mobile Application as the main developing application tool, to build the mobile application and gather all the necessary data extracted from given medical devices/wearables/IoT sensors raw/aggregated usage data associated to patient's activities, along with questionnaires completed by them and/or their carers during their whole participation to the project.

Garmin Health SDK 3.4.1

The Garmin Health SDK allows direct integration of Garmin wearables in Android or iOS applications. It gives direct access to all health and fitness activity data directly without the need for web service integration.

In the case of the RETENTION Mobile application, the Garmin Health SDK is used for the direct integration between Edge Mobile Application and Garmin smartwatches, since the project uses GARMIN Devices. In this instance, a seamless interaction between RETENTION Edge Mobile Application and the GARMIN tracking device, can continuously monitor and receive heart rate, step encounter and sleep quality for the registered patient.

² Bluetooth Low Energy: https://en.wikipedia.org/wiki/Bluetooth_Low_Energy

The Garmin Health SDK is not free for commercial use. A contract was signed between Garmin and Datamed, the partner developing the RETENTION Edge Mobile Application, dictating the terms and conditions under which the SDK is to be used, exclusively, for the purposes of the RETENTION research project.

Omron SDK

The OMRON SDK allows the RETENTION Edge Mobile Application to collect data from the OMRON devices to be used in the RETENTION clinical trial. Specifically the OMRON SDK is used for transferring the Oximeter, Pressure gauge and Weight scales data to the mobile application. The use of the SDK has been licensed by OMRON for the purposes of the RETENTION research project.

2.3 User journeys

The RETENTION Edge Mobile Application offer functionality to:

- The patients (and the carers)
- Technical staff at the patient's hospital (responsible for setting up the RETENTION devices for the patient)

Logging in

Upon opening the mobile application the login page appears. By clicking one of the two selections (patient or carer) the user can proceed either as a patient or a care giver. (The special admin option is reserved for device administrators and is not used by patients or carers).

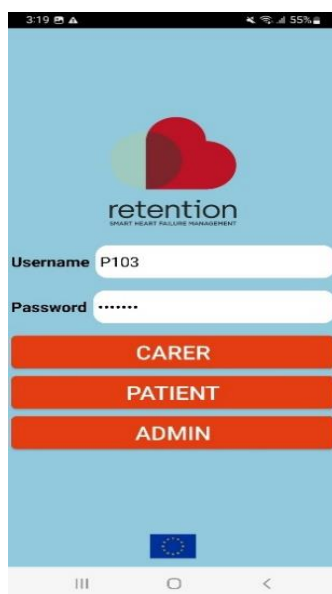


Figure 2: RETENTION mobile application login page

The main feature of the main screen, where the user lands upon logging in is the menu at the bottom.

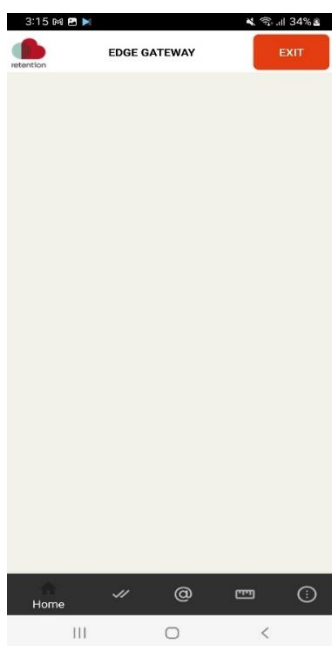


Figure 3: RETENTION mobile application main screen

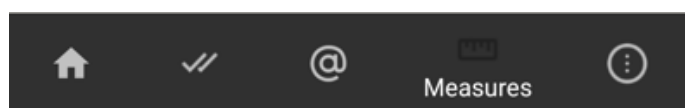


Figure 4: RETENTION mobile application menu tab

Obtaining Measurements

To navigate to the “Measures” screen, the user selects from the menu the corresponding option.

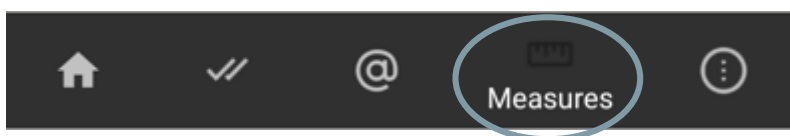


Figure 5: Measures option

The screen changes accordingly as shown below:

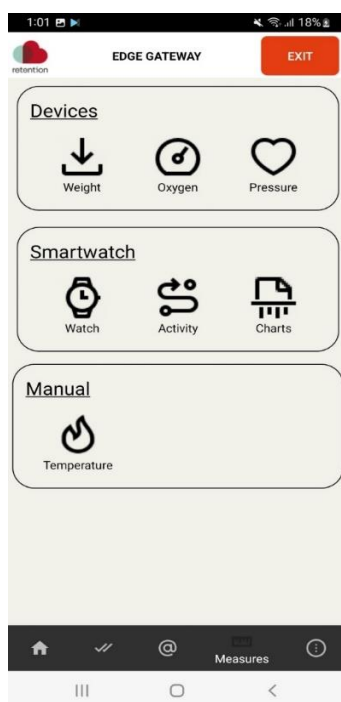


Figure 6: Measures screen

All measurements-related functionality, including device pairing is accessible from this screen. All related descriptions will assume it as the starting point. The user can return to it by pressing the “Measures” button as described above. The pairing processes are more complicated, that the subsequent data acquisition / display processes, but need not be performed again if pairing is already in place.

The individual functionalities described below support the RPT01 (patient data entry) and RTCA01 (carer data entry) Use Cases as defined in D3.1.

Pairing the patient’s smartwatch

In order for the smartwatch data to become available to the RETENTION Edge Mobile Application and subsequently to the patient’s hospital’s CSB, the smartwatch must first be paired with the mobile phone and the RETENTION Edge Mobile Application. If the watch is already paired, the patient/carers does not need to repeat this process.

To initiate the process of pairing the patient’s smartwatch with the RETENTION Edge Mobile Application, the user navigates to the ‘Measures’ screen if not there already and :

1. Taps on the Watch icon

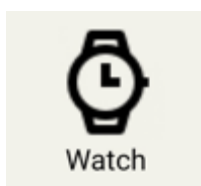


Figure 7: Smartwatch button

2. Taps on the 'Scan button that appears

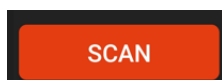


Figure 8: Scan button

3. Selects the patient's smartwatch from the list with the available devices to be connected



Figure 9: Available devices to be connected

4. Completes the information asked. (Height, weight, sleep time, wake up time)

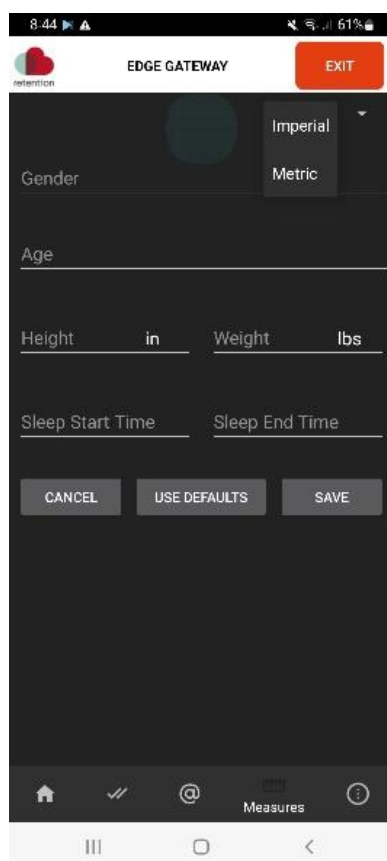


Figure 10: Personal information screen

5. Taps “Save” to save the data provided.

Syncing the patient’s smartwatch data

As the patient wears his/her smartwatch, the latter collects data even when the mobile phone is off or simply not in its vicinity. These data need to be sent into the RETENTION Edge Mobile Application, a process called syncing as it ensures the latter will have the most up-to-date data from the watch, so that they can then be made available to the CSB of the patient’s hospital. This is done wirelessly as long as the smartwatch is paired with the application and close to the phone.

To initiate the process, the user (patient/carer) navigates to the ‘Measures’ screen, if not there already, and:

1. Taps the ‘Paired Devices’ button



Figure 11: Paired devices button

The data from the watch will be automatically synced.

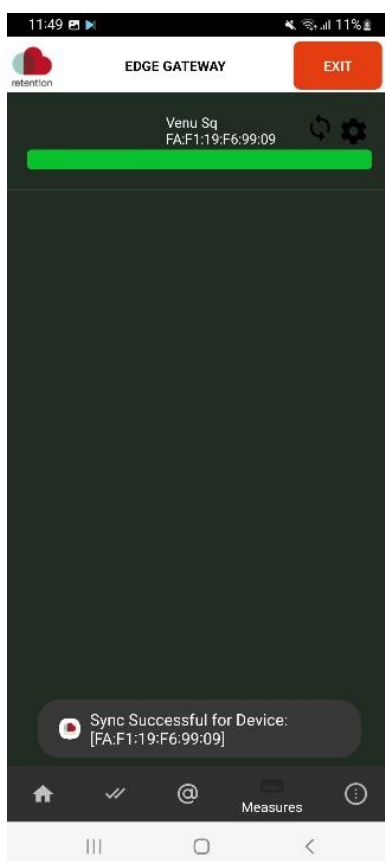


Figure 12: Synchronized device

Accessing patient activity charts

In order to access the Sleep, Step and Heart rate graphs, the user navigates to the 'Measures' screen, if not there already, and:

1. Taps on the 'Charts' option

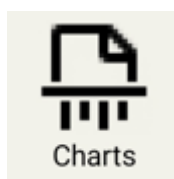


Figure 13: Charts button

2. Selects the chart option they are interested in.

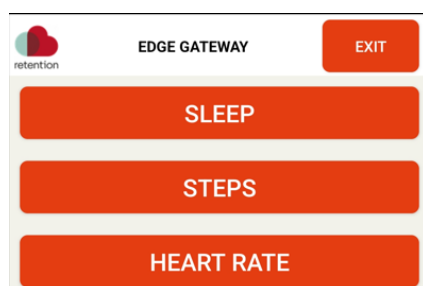


Figure 14: Available charts options

Assuming they wanted to get a chart of their heart rate, it would look similar to the example below:

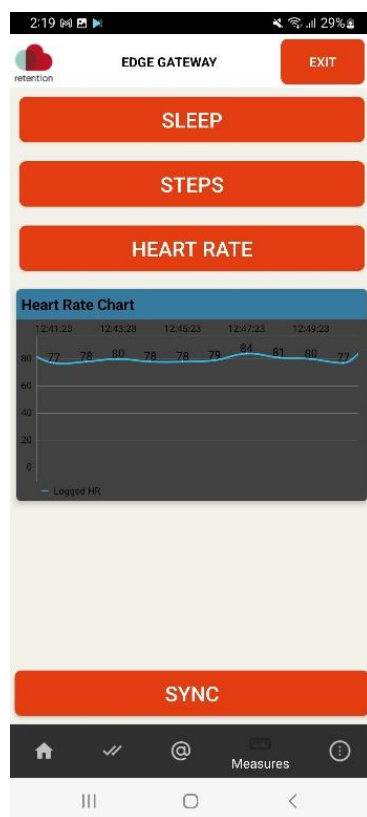


Figure 15: Heart rate chart example

Accessing recent activity metrics

In order to access the patient's recent activity metrics (number of steps taken, distance covered and calories burned), the user (patient/carer) navigates to the 'Measures' screen, if not there already, and:

1. Taps on the 'Activity' option.



Figure 16: Activity button

2. Taps on 'Sync' (to instruct the mobile application to get and display the very latest activity data from the smartwatch)

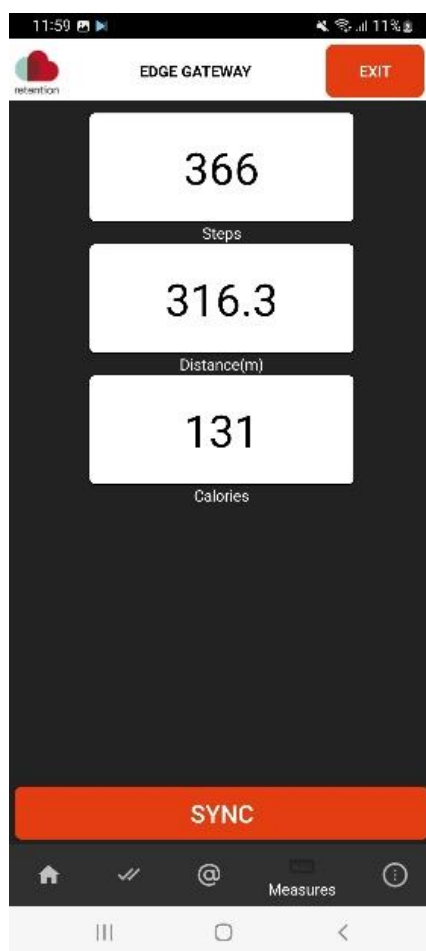


Figure 17: Activity results example

Pairing the patient's oximeter

It is necessary to pair the patient's oximeter with the RETENTION Edge Mobile Application, so that the later can obtain data from the oximeter and forward them also to the CSB of the patient's hospital. If the oximeter is already paired, the patient and/or the carer need not repeat the process.

To pair the oximeter, the user navigates to the 'Measurements' screen if not already there and:

1. Taps on 'Oxygen level' icon

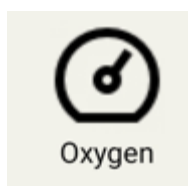


Figure 18: Oxygen level button

2. Taps on the 'Scan' button that appears



Figure 19: Scan button

3. Presses the oximeter power button for 3 seconds. The index 'P' will appear on the oximeter screen.

Once pairing is complete, syncing will be performed automatically and pulse and oxygen saturation values will be displayed on the RETENTION Edge Mobile Application.



Figure 20: Oxygen level result example

Syncing the oximeter data

In order to access the patient's oximeter readings, the user (patient/carer) navigates to the 'Measures' screen, if not there already, and:

1. Taps on 'Oxygen – level' icon

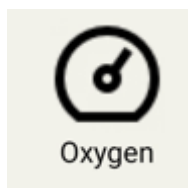


Figure 21: Oxygen level button

The oximeter measurements will be automatically synced.

The user can also see the history of the oximeter measurements by tapping on the 'PAST VALUES' button.

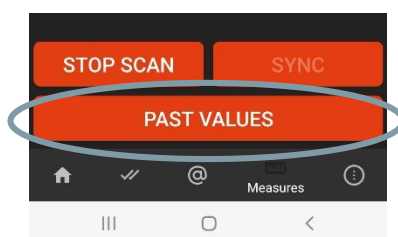


Figure 22: Measurement history button

Pairing the digital scales

It is necessary to pair the patient's digital scales with the RETENTION Edge Mobile Application, so that the later can obtain data from the scales and forward them also to the CSB of the patient's hospital. If the scales is already paired, the patient and/or the carer need not repeat the process.

To pair the digital scales, the user navigates to the 'Measurements' screen if not already there and:

1. Taps on 'Body weight' icon



Figure 23: Body weight button

2. Taps on 'Scan' button



Figure 24: Scan button

3. Presses the Bluetooth button on the digital scales for 5 seconds or until a P appears on the monitor

Unless a problem occurred, the scales is now paired. The most recent measurement is displayed in the mobile application.

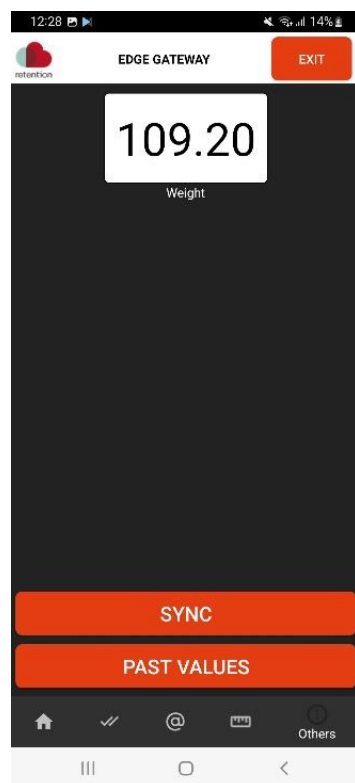


Figure 25: Weight measurement example

Syncing the digital scales data

In order to access the patient's digital scales readings, the user (patient/carer) navigates to the 'Measures' screen, if not there already, and:

1. Taps on 'Body weight' icon



Figure 26: Weight button

The patient's body weight measurements will be automatically synced. The history of the patients' weight measurements by clicking on 'PAST VALUES'

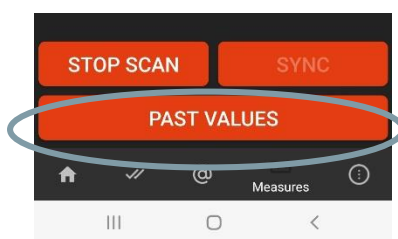


Figure 27: Measurement history example

Pairing the patient's digital blood pressure meter

It is necessary to pair the patient's digital pressure meter with the RETENTION Edge Mobile Application, so that the later can obtain the blood pressure data and forward them also to the CSB of the patient's hospital. If the scales is already paired, the patient and/or the carer need not repeat the process.

To pair the digital blood pressure meter, the user navigates to the 'Measurements' screen if not already there and:

1. Taps on 'Blood Pressure' icon

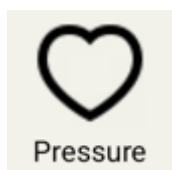


Figure 28: Blood pressure button

2. Taps on the 'Scan' button



Figure 29: Scan button

3. Presses the Bluetooth button on the digital blood pressure meter for 3 seconds or until the letter P appears on the monitor.

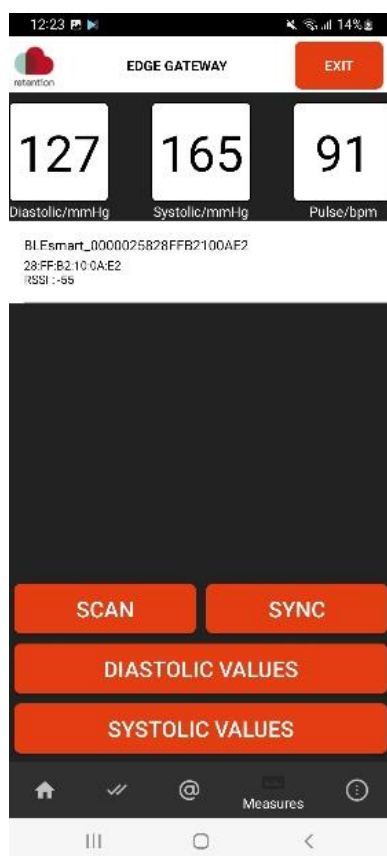


Figure 30: Blood pressure meter readings example

Syncing blood pressure meter data

In order to access the patient's digital blood pressure meter readings, the user (patient/carer) navigates to the 'Measures' screen, if not there already, and:

1. Taps on 'Blood Pressure' icon

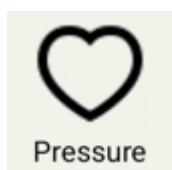


Figure 31: Blood pressure button

The patient's blood pressure measurements will be automatically synced.

The user can access the history of blood pressure measurements by clicking on 'DIASTOLIC/SYSTOLIC VALUES'

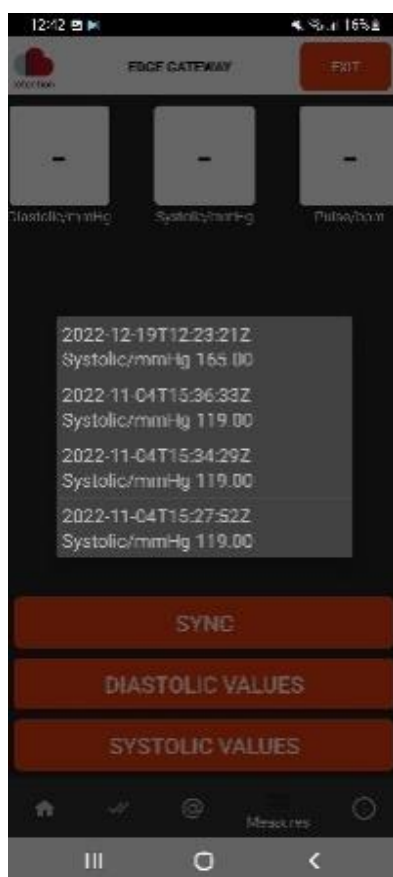


Figure 32: Blood pressure history measurements example

Body Temperature - Manual input

In addition to obtaining measurements automatically from paired devices, the RETENTION Edge Mobile Application allows patients/carers to enter certain values manually.

In order to enter body temperature readings, the user (patient/carer) navigates to the 'Measures' screen, if not there already, and:

1. Taps on the 'Temperature' button

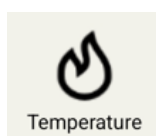


Figure 33: Temperature button

2. Enters the temperature

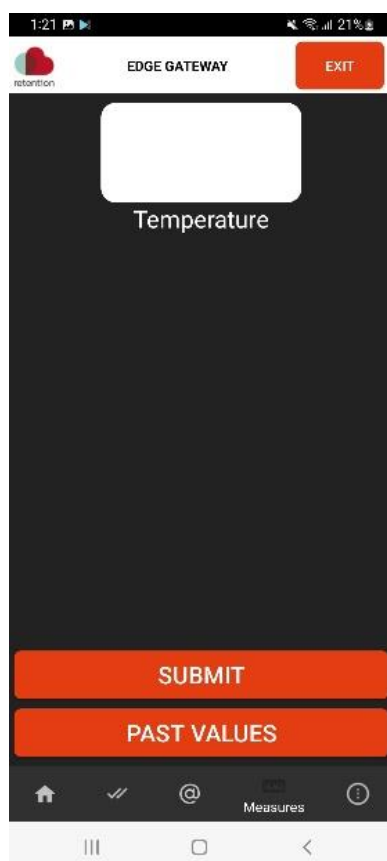


Figure 34: Temperature module main screen

3. Presses the “Submit” button.

In order to enter the VAD values, the user (patient/carer) :

1. Taps on the “Others” button on the bottom menu

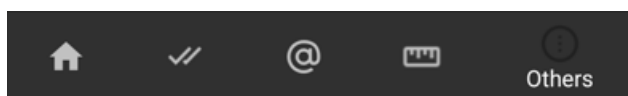


Figure 35: Menu tab

2. Taps on the VAD icon



Figure 36: VAD button

3. Manually inputs the VAD-reported values

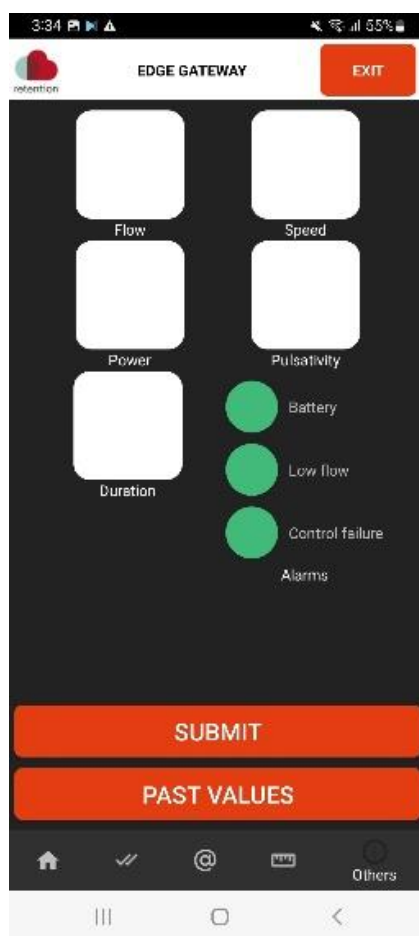


Figure 37: VAD module main screen

4. Presses the “Submit” button.

2.4 When to use it for each device

Data will be collected on a daily basis using the devices. Once a day, preferably in the morning, the patient (carer) should measure his or her body weight and blood pressure. Both measurements will be synced automatically as long as the pairing procedure is correctly performed. The oxygen saturation should be measured twice a day, in the morning and at the evening. Its measurement will automatically be synced likewise. Regarding the sleep activity, the data will be automatically synced, while heart rate and step encounter will continuously receive data. Following there is a table that summarises the time frame in which the patient (carer) shall perform each measurement.

Table 1: Timeframe of measurements

Weight	Once per day
Blood pressure	Morning - excluding patients with LVAD
Heart Rate	Continuous by wearing the Smartwatch
Peripheral capillary oxygen saturation	Morning, evening

Sleep: deep and light sleep phases, sleep interruptions	Continuous by wearing the Smartwatch
Steps, distance, floors climbed, calories consumed	Continuous by wearing the Smartwatch

Below is a diagram showing the flows of both automatic and manual measurements

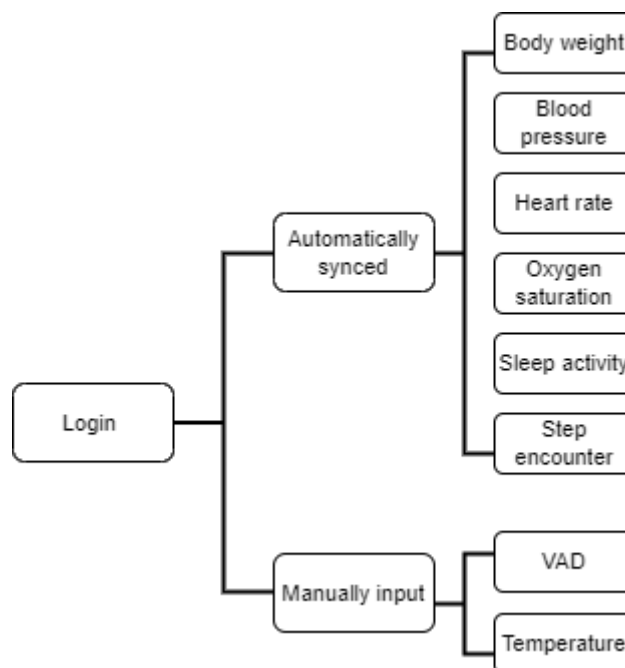


Figure 38: Measurement flow

2.5 Data flows

Inputs:

- The patient or career will interact with the system and will add local data as needed
- Data will flow from the devices to the mobile application (EDGE)

Outputs:

- FHIR medical data to each CSB via protected API
- Non-medical data to each CSB via protected API

Format:

- Inputs/outputs form and to the CSB in JSON format

2.6 Configuration details and APIs

Pre-specified subgroup analyses include:

- The three patient categories (HF, LVAD and HT)
- Males and Females

Data points:

- Oximeter (Oxygen saturation)



- Pressure gauge (Blood pressure, Heart rate) - excluding patients with LVAD
- Weight scales (Body weight)
- Smartwatch (pulse, Sleep & quality – Steps counter)
- VAD (Manual entry)
- Temperature (Manual entry)

Extra Measurements:

The following data will be additionally collected for patients with VAD:

- Controller parameters
- Alarms
- RPMs
- Power consumption
- Flow
- Pulsatility index

Devices – Equipment:

- Watch – Garmin
 - Garmin Venue SQ
- Oximeter – OMRON
 - Omron P300 Intelli IT pulse oximeter
- Weight scales -OMRON
 - Omron HN 300T2 Intelli IT
- Application edge (mobile device)
 - Samsung A03s
- Pressure gauge
 - Omron M4 Intelli IT

Alternative devices:

- Watch – Garmin
 - Garmin Venue SQ 2
- Application edge (mobile device)
 - Samsung A13
- Pressure gauge
 - Omron M7 Intelli IT

The communication between the CSB deployed components is done through REST APIs. Respectively, the CSB Security Component will authorise end-users and authenticate interactions between CSB components via REST services. Appendix 7.2 includes a detailed description of API data.



3 RETENTION Local Home Gateway

3.1 Description and objectives

The Local Home Gateway of the Patient Edge allows the seamless integration of smart devices and services with RETENTION platform. More specifically, it integrates a sensor (Xiaomi Mijia temperature and humidity sensor) and an external weather service to a Raspberry Pi computer, aiming at gathering, storing and uploading the non-FHIR (non-medical) data to the Global Insight's Cloud (GIC). These data consist of indoor and outdoor data such as temperature, humidity, external temperature, external humidity and air pollution index.

This local server should be capable of data handling, providing a user friendly and automated functionality. This system is developed in a user-friendly way to facilitate the users with minimal interaction. This device fetches the data wirelessly, making use of the BLE protocol regarding the temperature and humidity sensor, and an open-source weather service regarding the external condition data. More, specifically the main objectives and goals of the Local Home Gateway are the following:

- Real time measurement of temperature values in the participant's living place
- Real time measurement of humidity values in the participant's living place
- Real time measurement of external temperature values in the area of interest
- Real time measurement of external humidity values in the area of interest
- Real time measurement of pollution index in the area of interest
- Gathering of all the necessary values each hour (24 hours per day during the pilot)
- Local storage of these values
- Transmission of the data to the GIC through the non-FHIR API
- Handling of data loss and workaround solutions to minimise data loss
- Remote access ability for the technician to support the pilot process
- Identification and reporting of low sensor battery levels
- No user input, apart from power and ethernet connection
- Automatic updates making use of an open-source platform
- External weather and pollution data acquisition making use of zip codes, municipalities, areas etc.
- Transfer the data securely without any compromises, through the security component making use of specific credentials that refer to the Raspberry Pi local server and also the userId, the packageId and the organizationId.



Figure 39. The temperature and humidity sensor wirelessly connected to Raspberry Pi.

3.2 Data flow

The data flow is demonstrated by analysing two data flow/sequence diagrams that have been prepared to explain the following scenarios:

- The first scenario refers to the initial setup and configuration of the Local Home Gateway, that comprises a Raspberry Pi and a Xiaomi Mijia temperature and humidity sensor.
- The second one refers to the internal automatic process of retrieving the necessary data by the Local Home Gateway posting these non-FHIR data to the online database (GIC).

Before handing over the necessary equipment to the participant, the responsible technician of each pilot site, needs to configure a package of equipment with the necessary parameters and also test it locally at the hospital side, to make sure that the Local Home Gateway works as intended. The following paragraphs for data flow from local home gateway, as described in Figure 41. Data flow of retrieving data by the Local Home Gateway and posting to the GIC database., is presented below:

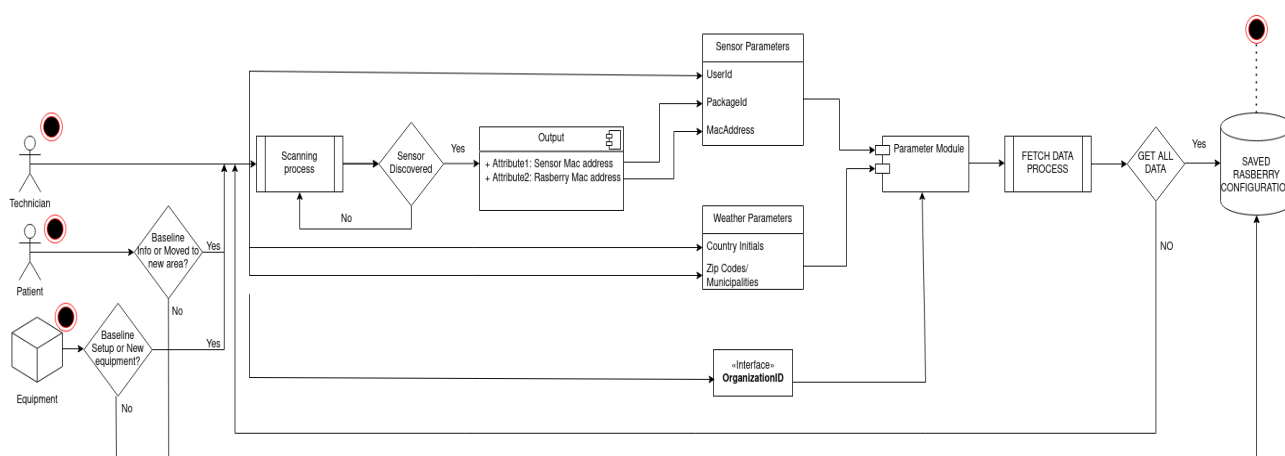


Figure 40: Data flow of the initial setup and configuration of the Local Home Gateway

- 1) The basic info needed is the package ID (mac address of the Raspberry Pi), the user ID (i.e. patient ID) and the zip code.
- 2) The technician initialises the scanning process to discover the Bluetooth sensor and pair it to Raspberry Pi.
- 3) The user ID, zip code and country code are then stored.
- 4) The organisation ID (corresponding to virtual pilot of pilots, OCSC, NKUA, SERMAS - HURC, SERMAS - HUPH, UNIBO, UKESSEN) is chosen by a drop-down list that includes all the pilot sites.
- 5) All the above information is stored in the parameter module.
- 6) A testing procedure fetches data from both sensor and external service.
- 7) If data fetch is successful, the configuration is stored in the Raspberry Pi workspace.

Based on the above, two cases are identified: In case the sensor is destroyed, this process has to be repeated by the technician. Similarly, in case the patient moves to another city, this process has to be repeated accordingly.

Having described the configuration data flow, the second scenario corresponds to the internal automatic process of retrieving the necessary data by the Local Home Gateway and posting these non -FHIR data to the online database.

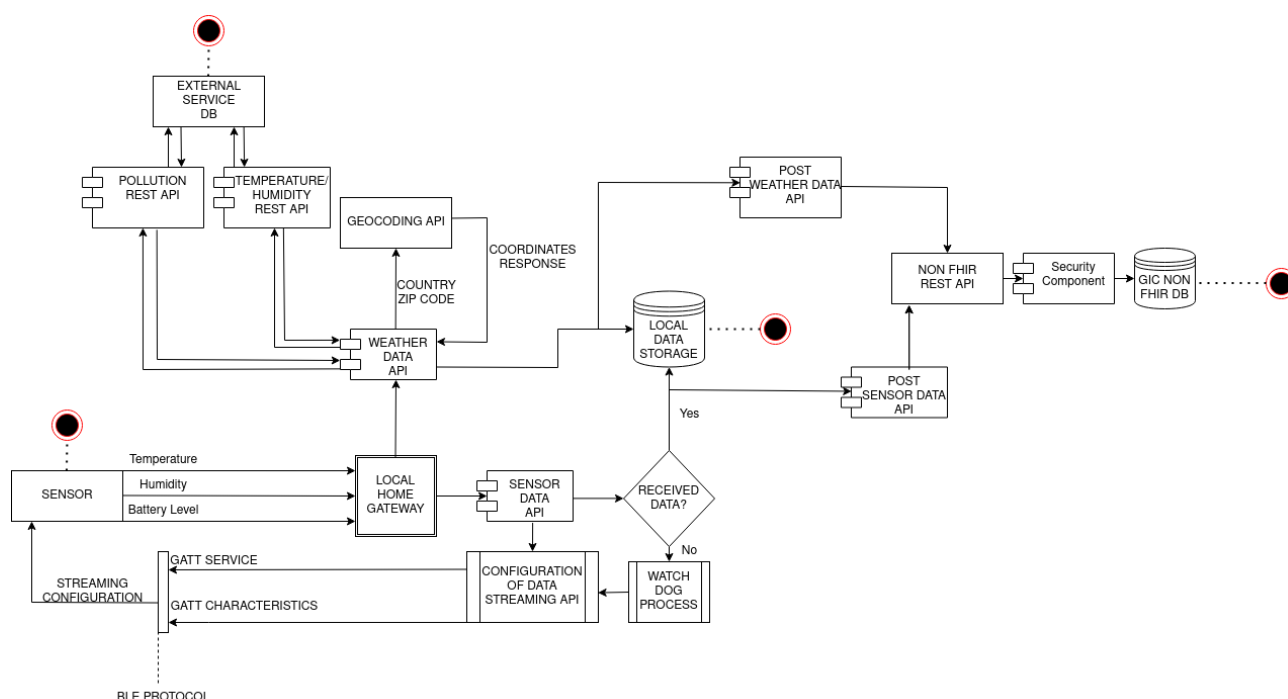
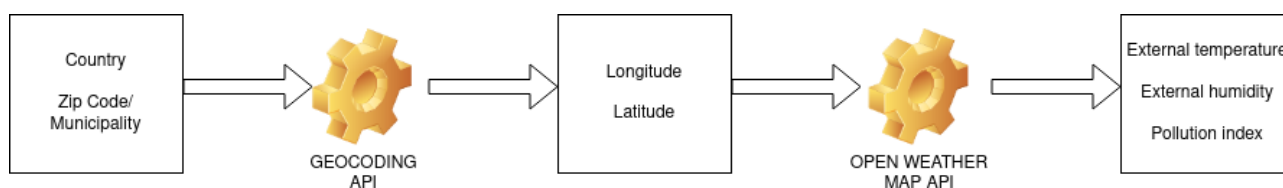


Figure 41. Data flow of retrieving data by the Local Home Gateway and posting to the GIC database.

The flow contains four functionalities:

- 1) Acquiring the sensor data: In one-hour interval the sensor is asked through the configuration of data streaming API to provide temperature, humidity and battery level values to the local home gateway.
- 2) Acquiring external temperature, humidity and pollution index (in one-hour interval): The first step is the conversion of zip code to coordinates (longitude and latitude), which is then used to trigger the temperature, humidity and pollution APIs of the Open Weather map service (<https://openweathermap.org/api>).



- 3) All the above values are stored in the local data storage.
- 4) After completed the above steps, 2 POST requests (for weather and sensor data) are executed, so that data is stored in GIC non-FHIR database.

The open weather service provides free access to the aforementioned required data, however, it uses the following authentication process. A registration to the online service is required via an email. For the context of the project, a general project email is used and will be used for all the participants as they don't need to have interaction with it.

After the registration, an online dashboard-like interface is provided to the registered user where he/she is able to generate as many API KEYS as he/she wishes. For the context of RETENTION, one API KEY for each patient is created and provided in the configuration files of the Raspberry Pi. After the generation of this API KEY, post and get requests are possible with the open weather APIs.

3.3 Device federation

In this section the details and format of the non-FHIR data and non-FHIR data structures are going to be described. In addition, the data preparation procedures will be explained and the .json files that are being created before posting the data will be presented.

As described previously, two are the main data sources streaming data, and that is the sensor and the external weather service. For both of them, the same implementation is in place, taking into consideration the data update frequency, as both the sensor and weather service are streaming data once each hour, by request³.

The instant values received from the two streaming points are the following:

Xiaomi Mijia temperature and humidity sensor data:

- Temperature in Celsius degrees. Indicates the temperature measured by the sensor at the living place of the patient. It is a float number with two decimal places. For example, a value received is 23.58°C.
- Humidity in percentage. Indicates the humidity measured by the sensor at the living place of the patient. It is an integer number with max value 100 (percentage). For example, a value received is 75%.
- Battery level in percentage. Indicates the sensor's life. It is an integer number with max value 100 (percentage). For example, a possible value received is 80%.

Weather Service data:

- External Temperature in Celsius degrees. Indicates the temperature as measured by the open weather service in the area that the corresponding patient currently lives in. It is a float number with two decimal places. For example, a value received is 23.58°C.
- External Humidity (percentage). Indicates the humidity as measured by the open weather service in the area that the corresponding patient currently lives in. It is an integer number with max value 100 (percentage). For example, a value received is 75%.
- Pollution index on a scale from 1-5. Indicates the air pollution or air quality as measured by the open weather map service in the area that the corresponding patient currently lives in. In the table below the pollution index in relation with the concentration of the pollutants is presented.

³ This frequency is configurable.

Qualitative name	Index	Pollutant concentration in $\mu\text{g}/\text{m}^3$			
		NO_2	PM_{10}	O_3	PM_{25} (optional)
Good	1	0-50	0-25	0-60	0-15
Fair	2	50-100	25-50	60-120	15-30
Moderate	3	100-200	50-90	120-180	30-55
Poor	4	200-400	90-180	180-240	55-110
Very Poor	5	>400	>180	>240	>110

Figure 42. Air quality index definition (source: <https://openweathermap.org/api>)

Taking into consideration that BLE protocol is in place, this configuration signal takes the form of the GATT Bluetooth protocol. GATT works by introducing many characteristics when establishing a Bluetooth connection; by establishing a BLE connection each time, a variety of GATT characteristics are available to use for the corresponding attribute that one wishes to obtain from the peripheral. In our case, we found out that to successfully obtain the necessary values (temperature, humidity and battery level) we need to utilise the characteristic with the UUID='EBE0CCBC-7A0A-4B0C-8A1A-6FF2997DA3A6' at the handle: 0x0036. The following figure presents a terminal output of the incoming data of seven consecutive calls of the above-described process.

```
Fetching data from A4:C1:38:DD:4A:79
Temperature: 22.4°C
Humidity: 65%
Battery: 92%

Fetching data from A4:C1:38:DD:4A:79
Temperature: 22.39°C
Humidity: 65%
Battery: 92%

Fetching data from A4:C1:38:DD:4A:79
Temperature: 22.39°C
Humidity: 66%
Battery: 92%

Fetching data from A4:C1:38:DD:4A:79
Temperature: 22.35°C
Humidity: 65%
Battery: 92%

Fetching data from A4:C1:38:DD:4A:79
Temperature: 22.33°C
Humidity: 65%
Battery: 92%

Fetching data from A4:C1:38:DD:4A:79
Temperature: 22.36°C
Humidity: 65%
Battery: 92%

Fetching data from A4:C1:38:DD:4A:79
Temperature: 22.28°C
Humidity: 65%
Battery: 92%
```

Figure 43. Terminal output of the incoming sensor's data.

These data then, as described in the previous section, are being converted in a .csv format and stored locally for preservation reasons. The following figure illustrates how these data are being preserved and hierarchically organised locally, in the workspace entitled by the package Id (a1:b2:c3:12:a2:b1) and the subdirectories one for each date the recordings take place.

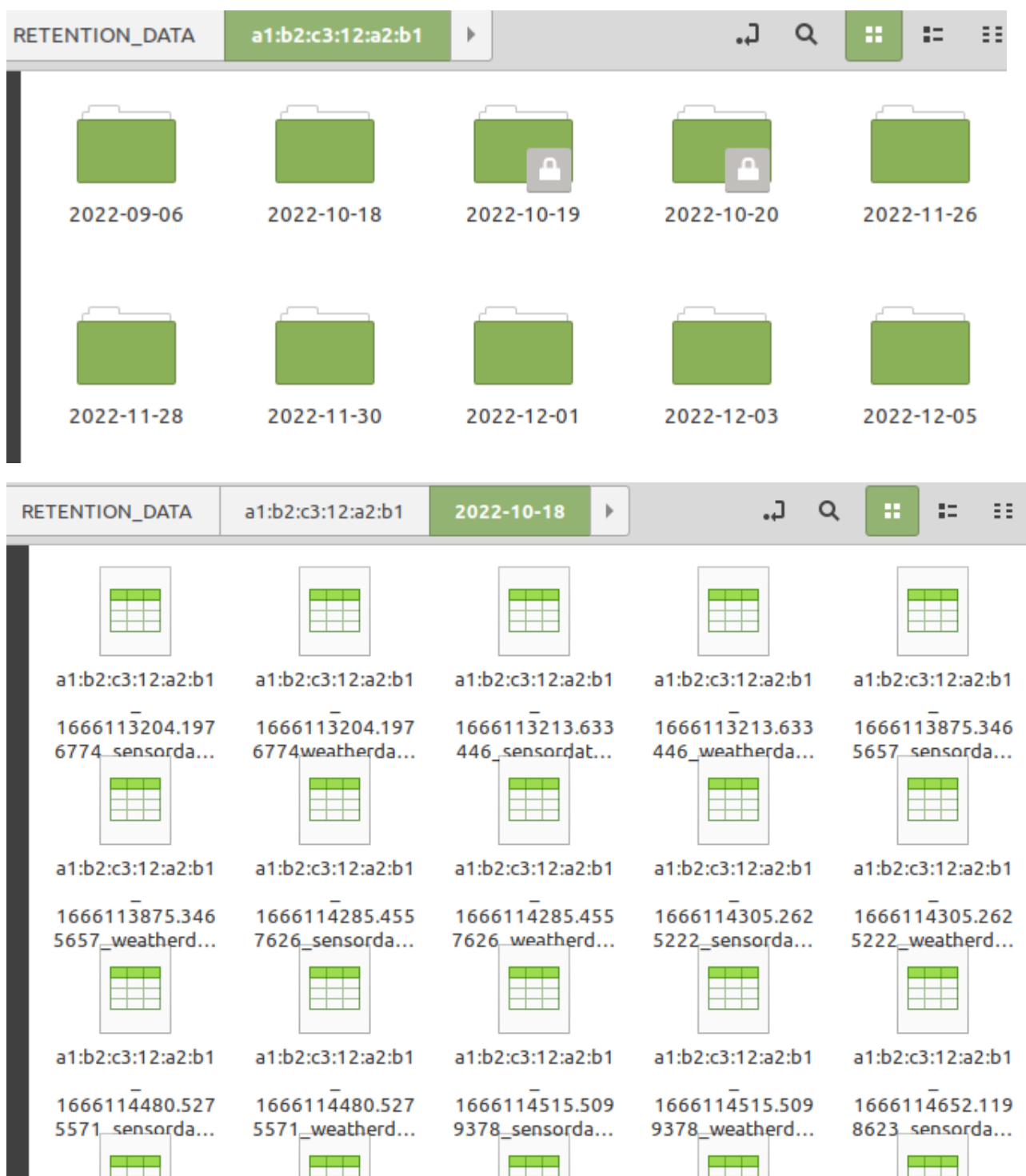


Figure 44. Local data storage structure.

Two types of .csv can be observed inside this local storage subdirectories, one referring to the sensor's raw data and the other referring to the open weather service's received data. The name convention that is being followed for these locally stored data in .csv formats, is: pagkageId_timestamp_sensordata/weatherdata.csv.

It should be noted that timestamping is applied. The timestamp is reported in seconds in epoch/unix time.

After the local saving of the data, they should be prepared accordingly, to be posted and saved on the GIC non- FHIR database. So the preparation of data is necessary before posting that has a specific .json format that needs to be followed. Regarding the sensor data, one .json for each variable should be created before posting. The .json format for the temperature, humidity and battery level variables is depicted in Appendix section

Next the process of obtaining the open weather data, as well as their format and data structure format as the posting of data is concerned, will be explained and described.

The first thing that needs to be done in order to begin the process of obtaining the external weather data is the generation of API keys. The open weather service for security reasons requires an API key in each of the post requests that take place. In order to generate an API key an account needs to be created with some information to be provided to the open weather service and an email account. After that one can generate as many API keys as he/she likes, to use with the free provided open weather APIs. Currently 5+1 API keys have been generated for the needs of the 5 pilot centers+the VPoP. In potential future applications of RETENTION, where addition centers join, additional API keys will generated. The free APIs can cover the needs of the RETENTION project as they provide the instantaneous required parameters by request. The following screen presents the generation of API keys screen.

Key	Name	Status	Actions	Create key
e58eada8a54a3441a6d8162fb2ae8b1b	Default	Active	  	API key name <button>Generate</button>
7b3bed5681941183a428898c12c7a2a9	retentionapi	Active	  	
8eeb1699712cf88608c9cea4d15a082d	retentionapi2	Active	  	
80594faaec17b29647004a29bec41add	retentionapi3	Active	  	
c0f5413dc0ea821ec4153bc0a356af7b	retentionapi4	Active	  	
b4f73f8f87b76a8f43654a1aa4cb8d61	retentionapi5	Active	  	

Figure 45. The generation of API keys.

The next step of the software is to call the first open weather map API, which is the geocoding API to convert the inserted variables that are zip code and country to the familiar with the open weather API variables, longitude and latitude. This API call is presented next.

API call

```
http://api.openweathermap.org/geo/1.0/zip?zip={zip  
code},{country code}&appid={API key}
```



Figure 46. Open weather map API calls.

In the following figure geocoding API responses are presented, when triggering the API with a zip code for Bologna, Italy and Essen, Germany.

```
{"zip": "40121", "name": "Bologna", "lat": 44.4938, "lon": 11.3387, "country": "IT"}  
{"zip": "45127", "name": "Essen", "lat": 51.4535, "lon": 7.0102, "country": "DE"}
```

Figure 47. Geocoding API responses.

Next the API call in order to retrieve the current weather data for the longitude and latitude provided, is presented.

API call

```
https://api.openweathermap.org/data/2.5/weather?lat=  
{lat}&lon={lon}&appid={API key}
```



Figure 48. API call to retrieve weather data based on longitude and latitude.

The query parameters in order to establish a successful get request are latitude, longitude and the API key and were explained previously. The response of a successful call is as follows.

```
{"coord":{"lon":11.3426,"lat":44.4949},"weather":[{"id":701,"main":"Mist","description":"mist","icon":"50n"}],"base":"stations","main":{"temp":279.41,"feels_like":278.48,"temp_min":277.93,"temp_max":285.27,"pressure":1015,"humidity":95},"visibility":1700,"wind":{"speed":1.54,"deg":330},"clouds":{"all":75},"dt":1670358903,"sys":{"type":2,"id":2000038,"country":"IT","sunrise":1670308574,"sunset":1670340917},"timezone":3600,"id":3181928,"name":"Bologna","cod":200}
```

Figure 49. Response of the API call.

As the service temperature in Kelvin degrees a conversion is applied.

The final information that needs to be retrieved is the pollution/air quality index and so the final open weather map API call is presented next, to the corresponding endpoint.

API call

```
http://api.openweathermap.org/data/2.5  
/air_pollution?lat={lat}&lon={lon}&appid={API key}
```

Figure 50. API call to retrieve the air pollution data.

The next figure displays the resulted .json file that is received when establishing a get request using the air pollution API. As per clinical users' advice, only the general air quality index is kept and not the specific pollutants.

```
{"coord":{"lon":11.3387,"lat":44.4938},"list":[{"main":{"aqi":4},"components":{"co":1174.93,"no":27.05,"no2":26.05,"o3":0,"so2":2.06,"pm2_5":45.41,"pm10":48.56,"nh3":8.11},"dt":1670361910}]}
```

Figure 51. Response of the API call for the air pollution.

Based on the above information, the relevant .json files are prepared for posting to the GIC database (Figure 52).

```
externalTemperature.json x
{
  'userId': 'P103',
  'serialNumber': 'a1:b2:c3:12:a2:b1\n',
  'measurementItems':
  [
    {
      'state': '14.560000000000002',
      'type': 'temperatureExternal',
      'deviceId': 'rpi',
      'measurementDatetime': 1670437855
    }
  ]
}

externalHumidity.json x
{
  'userId': 'P103',
  'serialNumber': 'a1:b2:c3:12:a2:b1\n',
  'measurementItems':
  [
    {
      'state': '80',
      'type': 'humidityExternal',
      'deviceId': 'rpi',
      'measurementDatetime': 1670439795
    }
  ]
}

pollutionIndex.json x
{
  'userId': 'P103',
  'serialNumber': 'a1:b2:c3:12:a2:b1\n',
  'measurementItems':
  [
    {
      'state': '1',
      'type': 'pollutionIndex',
      'deviceId': 'rpi',
      'measurementDatetime': 1670440120
    }
  ]
}
```

Figure 52. The .json format of external temperature, humidity and pollution index variables.

3.4 Configuration details and APIs

The following section summarises the technician tools and the relevant APIs. Below, the configuration screen that a technician can use in order to pre-configure the Raspberry PI before delivering it to the patient, is presented (Figure 53). It is noted, that for the configuration process, the Raspberry PI should be connected to an external monitor, and after booting, the following configuration panel will be available.

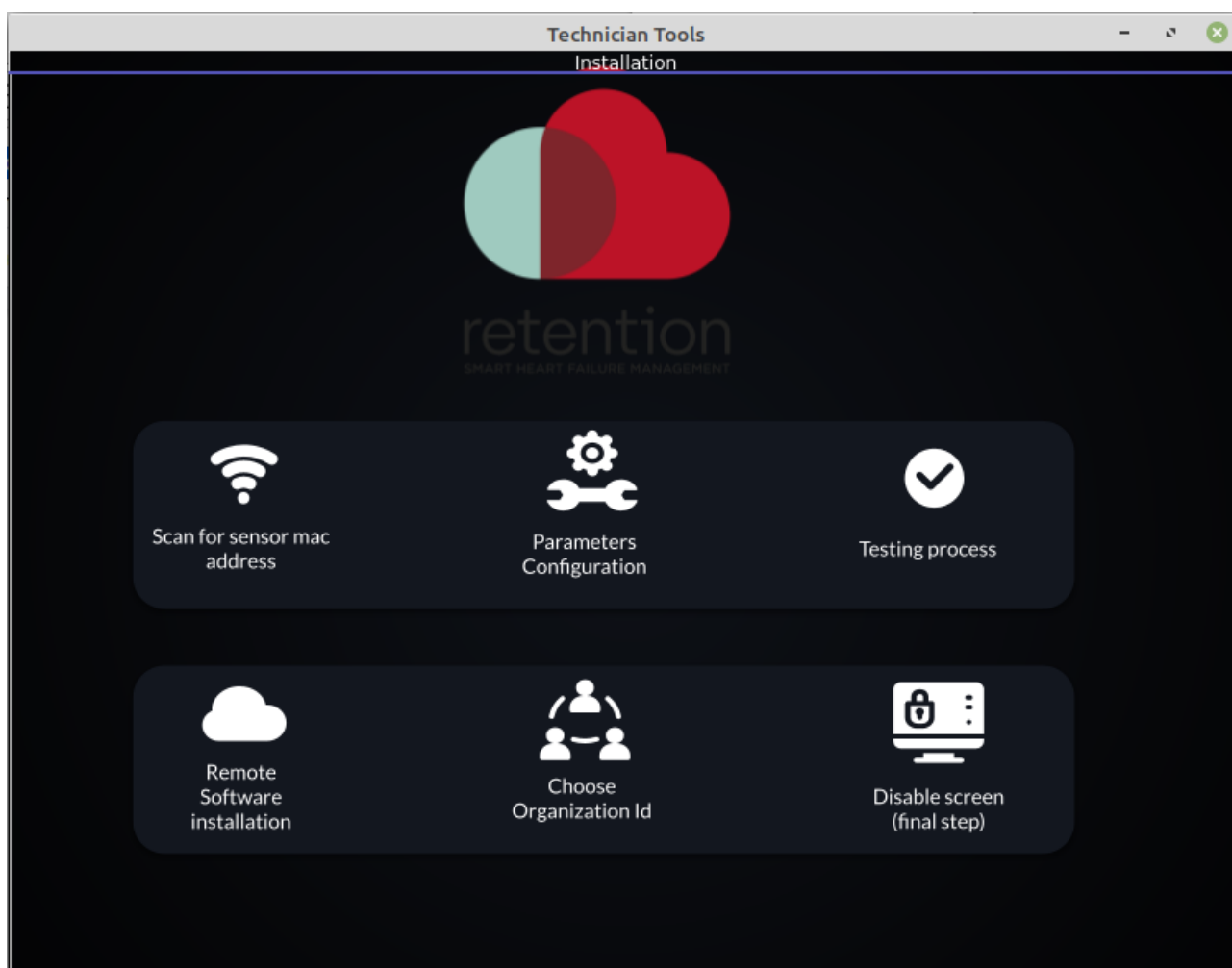


Figure 53. The technician tools for the Local Home Gateway setup.

Six options are presented to the user:

- 1) Scan for sensor mac address to retrieve sensor's mac address and package ID
- 2) Parameters configuration: (a)The userId (b)The package Id (c)The Mac Address, (d) Country and (e) Zip Code
- 3) Testing process to check if the configuration is successful and data is properly retrieved
- 4) Remote software installation for installing remote access software for potential remote troubleshooting
- 5) Organisation ID for selecting among the different clinical sites
- 6) Disable screen, which disables the Raspberry Pi screen.

SENSOR DATA API

The following figure provides a pseudocode for the sensor data API functionality.

```
Read configuration_file
Initialize variable mac_address
Initialize variable user_Id
Initialize variable package_Id
Initialize variable endpoint_url
Call function configuration_api with input mac_address
configuration_api returns variables temperature,humidity,battery_level
if configuration_api successful
    update flag_received_data to true
    if not (directory package_Id exists)
        create directory package_Id
    change directory to package_Id
    if subdirectory current_date exists
        change directory current_date
        convert json to csv
        save csv
    else
        create directory current_date
        change directory current_date
        convert json to csv
        save csv
    Call function post_sensor_data with input temperature,humidity,battery_level
else
    update flag_received_data to false
```

Figure 54. Pseudocode for the sensor data API functionality.

CONFIGURATION API

The following figure provides pseudocode for the configuration API functionality.

```
Initialize variable characteristic to 'EBE0CCBC-7A0A-4B0C-8A1A-6FF2997DA3A6'
Write value 0100 to characteristic
Receive data stream of values temperature,humidity,battery_level
Return temperature,humidity,battery_level to SensorDataApi
```

Figure 55. Pseudocode for the configuration API functionality.

WATCHDOG PROCESS

A watchdog process is running with a timer of 10 minutes, by making use of the crontab schedule tool provided by the Linux operating system. This process has the task of finding out if the communication between the sensor was successful and the data are retrieved without a problem, and if not to re-establish the whole procedure of receiving the sensor's data.

The following figure provides a pseudocode for the watchdog process.

```
# Runs every 10 minutes
Read flag_received_data
if flag_received_data is false
    Call function SensorDataApi
```

Figure 56. Pseudocode for the watchdog process.

POST SENSOR DATA API

The following figure provides pseudocode for the post sensor data API.

```
Initialize variable username
Initialize variable password
Initialize token_url
Receive variables temperature, humidity, battery_level from SensorDataApi
Call function post_request with input username, password, token_url
Receive token from post_request
Initialize non_fhir_url
Convert temperature to temperature.json
Convert humidity to humidity.json
Convert battery_level to battery.json
Call function post_request with input temperature.json, token, non_fhir_url
Call function post_request with input humidity.json, token, non_fhir_url
if battery_level less than 10%
    Call function post_request with input battery.json, token, non_fhir_url
```

Figure 57. Pseudocode for post sensor data API functionality.

NON FHIR REST API

The non-FHIR REST API (developed in the framework of **WP4**) is being utilised by posting to the correct non-FHIR endpoint, using as headers the access token received by the security component, and converting the received sensor data in the appropriate JSON format as explained in the previous chapter. This API is provided to the Raspberry Pi and so apart from the aforementioned parameters no other development is needed.

WEATHER DATA API

The following figure provides pseudocode for the weather data API functionality.

```
Read configuration_file
Initialize variable city
Initialize variable country
Initialize variable zip_code
Initialize variable user_Id
Initialize variable package_Id
Initialize variable endpoint_url
Initialize weather_api_key
Call function geocoding_api with input country,zip_code,weather_api_key
Receive parameters longitude,latitude from geocoding_api
Call function external_weather_api with input longitude,latitude,weather_api_key
Receive external_weather.json
Convert external_weather.json to variables external_temperature,external_humidity
Call function air_pollution_api with input longitude,latitude,weather_api_key
Receive air_pollution.json
Convert air_pollution.json to variable pollution_index
if not (directory package_Id exists)
    create directory package_Id
    change directory to package_Id
    if subdirectory current_date exists
        change directory current_date
        convert json to csv
        save csv
    else
        create directory current_date
        change directory current_date
        convert json to csv
        save csv
Call function post_weather_data with input external_temperature,external_humidity,pollution_index
```

Figure 58. Pseudocode for weather data API functionality.

In addition, the following open weather map APIs have been utilised:

- GEOCODING API
- TEMPERATURE AND HUMIDITY API
- POLLUTION INDEX API

POST WEATHER DATA API

The following figure provides pseudocode for the post weather data API functionality.

```
Initialize variable username
Initialize variable password
Initialize token_url
Receive variables external_temperature,external_humidity,pollution_index from WeatherDataApi
Call function post_request with input username,password,token_url
Receive token from post_request
Initialize non_fhir_url
Convert external_temperature to external_temperature.json
Convert external_humidity to external_humidity.json
Convert pollution_index to pollution_index.json
Call function post_request with input external_temperature.json,token,non_fhir_url
Call function post_request with input external_humidity.json,token,non_fhir_url
Call function post_request with input pollution_index.json,token,non_fhir_url
```

Figure 59. Pseudocode for the post weather data API functionality.



MEMORY HANDLING PROCESS

Addition functionality has been developed for optimising memory handling. The following figure provides pseudocode for the memory handling API.

```
# Running each hour
Scan for open python processes
Return list of processes
Terminate open processes and free memory
Scan for open terminal processes
Return list of open terminal processes
Terminate open processes and free memory
```

Figure 60. Pseudocode for the memory handling API.

In order for the system to run, the following tools and libraries are used:

- Python 3
- Anydesk remote software (in cases the patients opts-in for remote problem solving)
- requests version 2.22.0
- Json version 0.9.5
- lywsd03mmc version 0.1.0
- PyQt5 version 5.11.3
- Pyqtgraph version 0.13.1
- Numpy version 1.22.3
- Bluez version 5.53

The Installation instructions are given in detail in the **Appendix 7.1**.

4 RETENTION Clinical Site Backend Dashboard & Visualisations

This section presents the main functionalities supported by the CSB and the clinician view of the Dashboard part, where the physician can monitor patients' data and can also analyse patients' evolution, with help of the provided AI tools integrated in the CSB Visualisation.

4.1 Description and objectives

The CSB aims to provide the clinician with a GUI where he or she can find, process and analyse the data gathered by the RETENTION platform for all patients. Also, clinicians will be able to load pre-trained models, and visualise the general context of patient's evolutions, influence of medication or lifestyle to their health, so they can easily take decisions supported by mathematical tools.

The current version of the CSB provides various functionalities and provides the visualisation means for the following uses cases, described in detail in D3.2.

Clinician's use cases

- **RTCL01** Creation of patient and data visualisation: Clinical team checks patients' data and alarms
- **RTCL02** Patient-doctor interactions: enter and store the information during a patient visit
- **RTCL03** Alarms: Important patients' alarms and detected decompensations
- **RTCL04** Event record: every event occurring to the patient

Risk assessment generated by AI

- **RTBD01** Risk assessment generated by AI

Technical use cases

- **RTS01** Dashboard End-user moderated registration
- **RTS02** Dashboard End-user login
- **RTS03** Manage end-user account information
- **RTS04** Associating a device ID to a patient

The RETENTION CSB consists of a collection of components placed at the clinical locations in charge of the health monitoring and wellbeing of the HF patients who are enrolled for the program. Each of these clinical sites has a unique RETENTION CSB instance that is exclusively focused on patient data ingestion (i.e., real-time prospective data) as received from the RETENTION Edge instances of the monitored patients. This data, when combined with clinical (retrospective and periodically updated) data, enable the execution of local analytics based on verified interventions models.

4.2 Dashboard tools and libraries

The CSB Dashboard & Visualisation are based on the Advanced Visualisation Toolkit (AVT) of AEGIS and will be deployed at the various clinical sites. AVT frontend is based on Angular.io framework (a TypeScript-based, and open-source web application framework) in conjunction with a set of visualisation libraries that are interchangeably used to cover specific needs and particularities of the analysed data. These include interactive timeline representations for time-dependent data, graph-based visualisations, and all common types of charts and graphs to communicate data points with impact e.g. bar chart, line graphs, scatter plot, pie charts etc.

In terms of data storage capabilities, AVT can support a variety of storage facilities (Figure 61) and can fully support the storage technologies defined in RETENTION architecture (see Del. 3.2). This flexibility is possible due to a number of tailor-made converters, which can request the data from the respective data sources and

transform it to a suitable format for AVT's visualisation widgets. These transformations together with internal configuration options and business logic activities take place in AVT's middleware application built on Node.js.

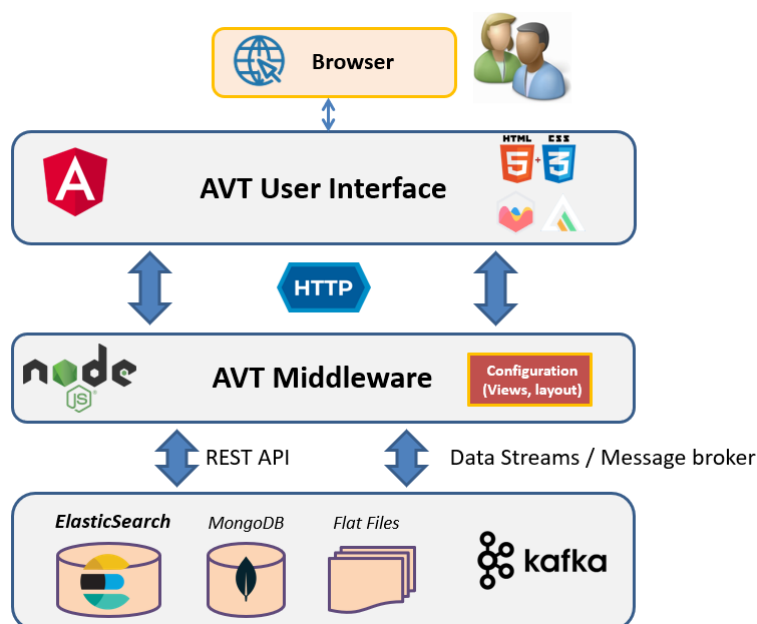


Figure 61: AVT Technology Stack

The AVT can be also used by users to trigger actions. These actions will mostly include selection of data sources and time periods. The data retrieval will take place using one of the aforementioned methods that will be available in the project.

The minimum hardware Requirements are summarized below:

- CPUs: ≥ 2
- RAM: ≥ 4 GB (depending on the data volume and utilised visualisations)
- Disk space: ≥ 20 GB (depending on the data volume and utilised visualisations)

4.3 User journeys

Based on the user requirements and the desired functionality, a range of different use cases and scenarios are supported by the CSB and are described in the following subsections.

The front page of the CSB is depicted in the following figure (Figure 61). The layout has 3 main areas (A, B and C). Area 'A' is the top menu with the login info of the user, Area 'B' where the vertical menu is lied and also various blocks of information are presented (e.g. Notifications) and Area 'C' which is the main area of presenting info, figures and monitoring results.

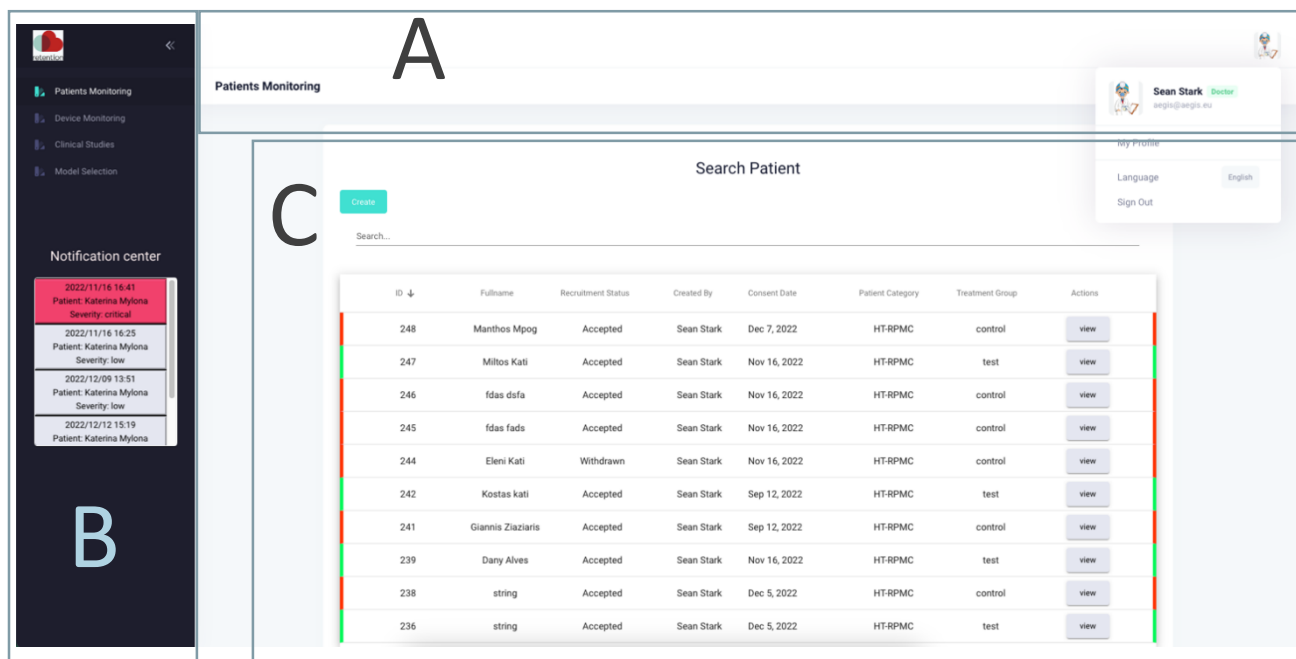


Figure 62: CSB Frontpage for the Role 'Clinician'

4.3.1 Clinician's use cases

RTCL01 Creation of patient and data visualisation: Clinical team checks patients' data and alarms

From the vertical menu 'Patients Monitoring' the user can access the complete list of the patients of the specific CSB. The user (e.g. clinician) can search for a specific patient by using various criteria such as full name, ID, date, status, clinician who created the user etc.

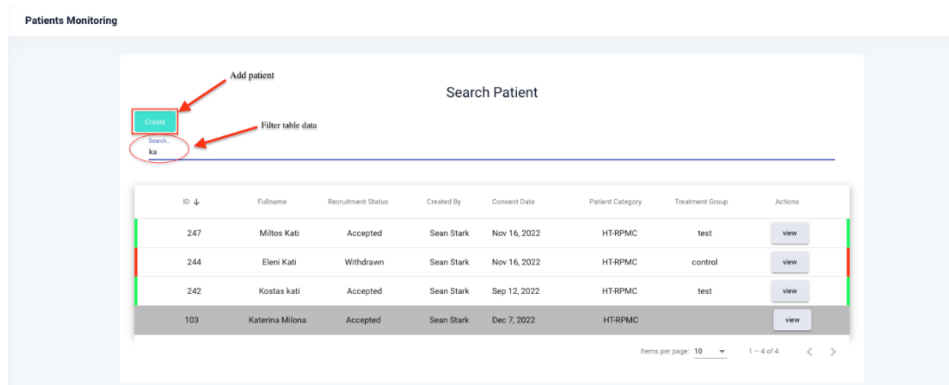


Figure 63: Search for specific patient

The 'create' button leads to the following screen where all minimum required data should be entered.

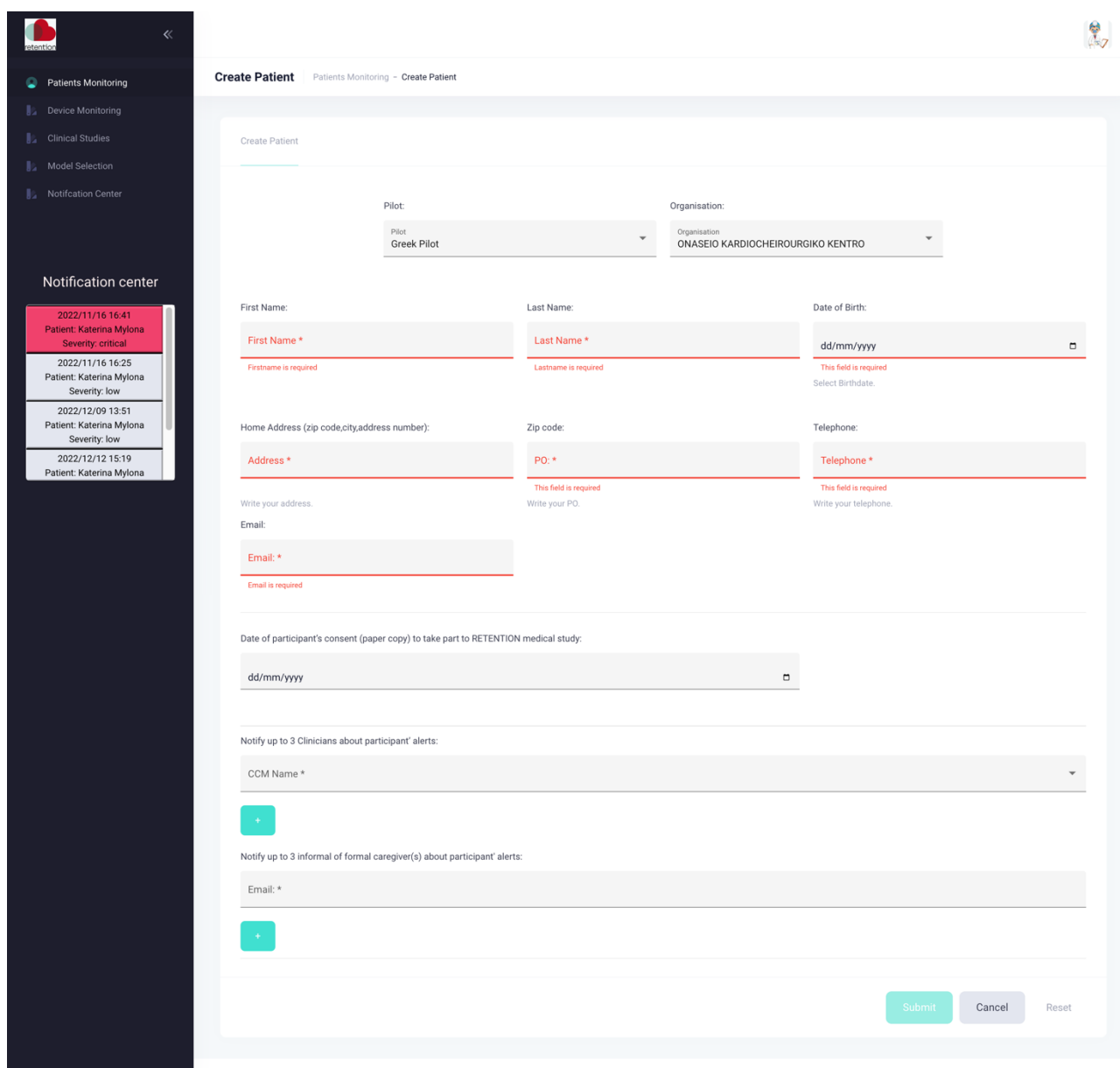
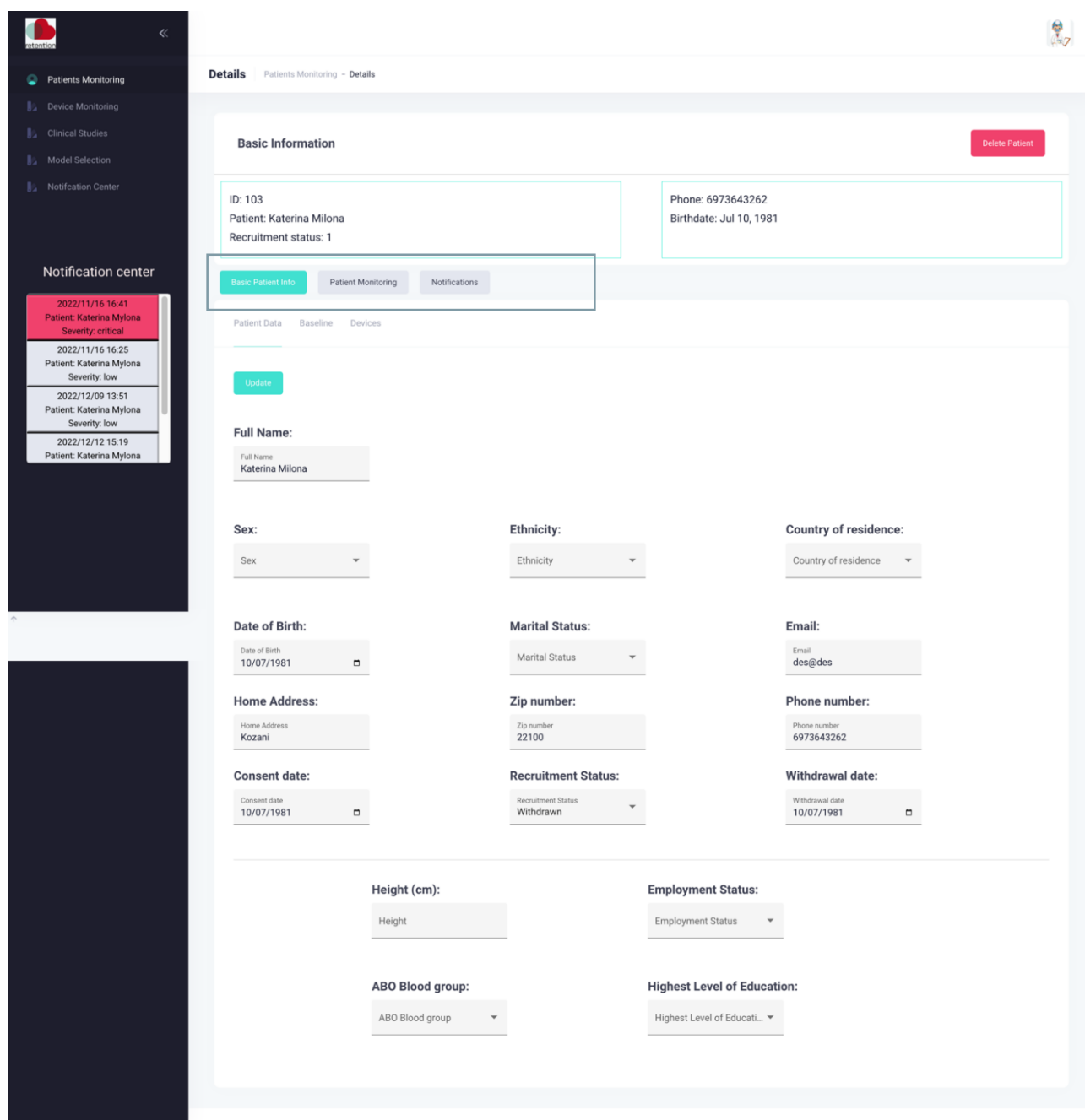


Figure 64: Patient creation screen

After the creation of the patient, the clinician can update the profile of the specific patient and store and track additional info, organized in 3 tabs: 'Basic Patient Info', 'Patient Monitoring' and 'Notification'. The user can update the patient record accordingly.



The screenshot displays the 'Details' page for a patient named Katerina Mylona. The interface includes a sidebar with navigation options: Patients Monitoring, Device Monitoring, Clinical Studies, Model Selection, and Notification Center. The main content area is titled 'Details' and 'Patients Monitoring - Details'. It features a 'Basic Information' section with fields for ID (103), Patient Name (Katerina Mylona), Recruitment status (1), Phone (6973643262), and Birthdate (Jul 10, 1981). Below this is a 'Patient Monitoring' tab with an 'Update' button. The 'Full Name' field is filled with 'Katerina Mylona'. Other fields include Sex, Ethnicity, Country of residence, Date of Birth (10/07/1981), Marital Status, Email (des@des), Home Address (Kozani), Zip number (22100), Phone number (6973643262), Consent date (10/07/1981), Recruitment Status (Withdrawn), Withdrawal date (10/07/1981), Height (cm), Employment Status, ABO Blood group, and Highest Level of Education.

Notification center

- 2022/11/16 16:41
Patient: Katerina Mylona
Severity: critical
- 2022/11/16 16:25
Patient: Katerina Mylona
Severity: low
- 2022/12/09 13:51
Patient: Katerina Mylona
Severity: low
- 2022/12/12 15:19
Patient: Katerina Mylona

Details Patients Monitoring - Details

Basic Information Delete Patient

ID: 103
Patient: Katerina Mylona
Recruitment status: 1

Phone: 6973643262
Birthdate: Jul 10, 1981

Basic Patient Info Patient Monitoring Notifications

Patient Data Baseline Devices

Update

Full Name:
Full Name
Katerina Mylona

Sex:
Sex

Ethnicity:
Ethnicity

Country of residence:
Country of residence

Date of Birth:
Date of Birth
10/07/1981

Marital Status:
Marital Status

Email:
Email
des@des

Home Address:
Home Address
Kozani

Zip number:
Zip number
22100

Phone number:
Phone number
6973643262

Consent date:
Consent date
10/07/1981

Recruitment Status:
Recruitment Status
Withdrawn

Withdrawal date:
Withdrawal date
10/07/1981

Height (cm):
Height

Employment Status:
Employment Status

ABO Blood group:
ABO Blood group

Highest Level of Education:
Highest Level of Educati...

Figure 65: Patient's basic info fields

The 'Patient Monitoring' tab (see below) presents all the available monitoring data for this specific patient, as captured by the monitoring devices.

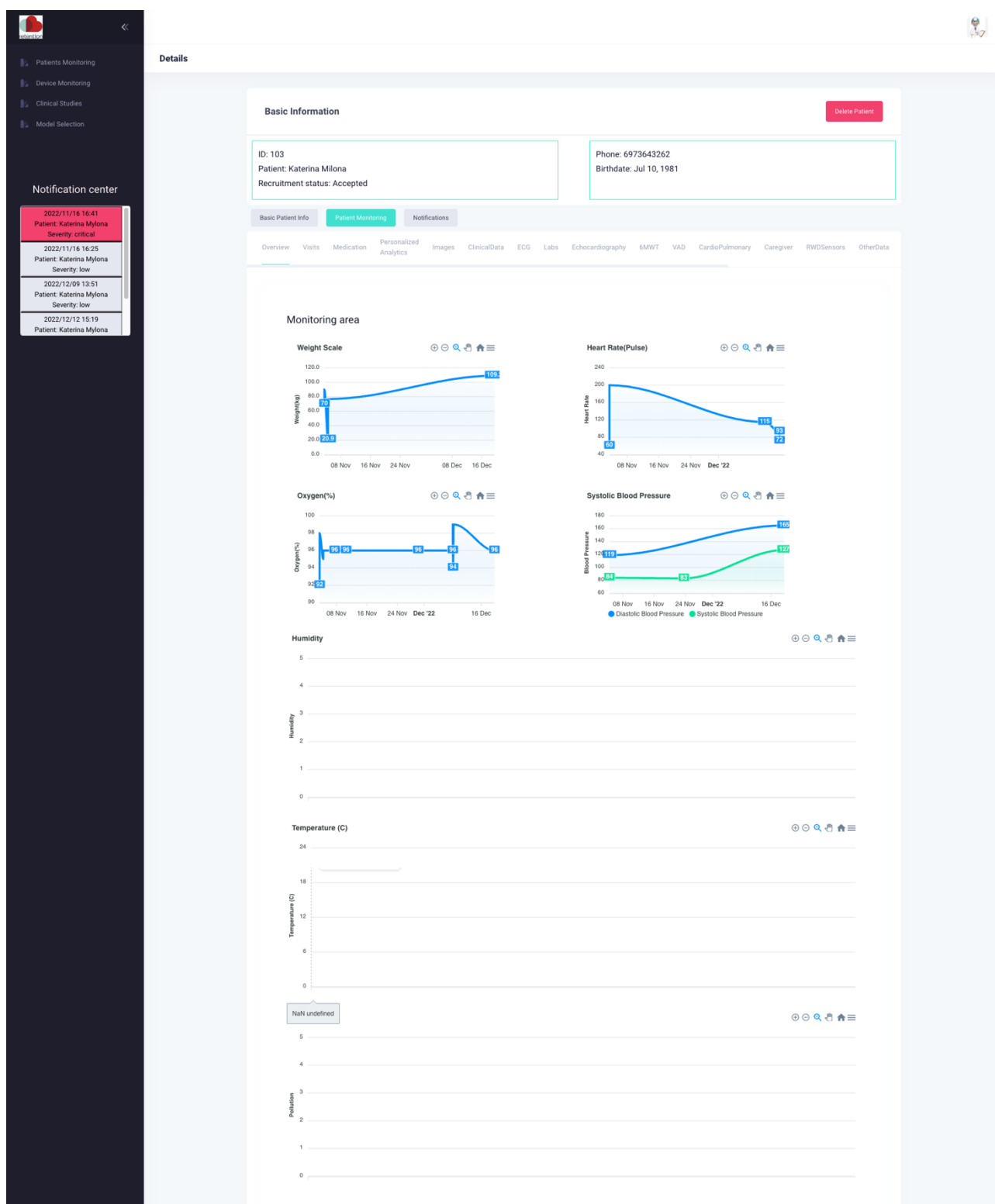


Figure 66: Patient monitoring screen

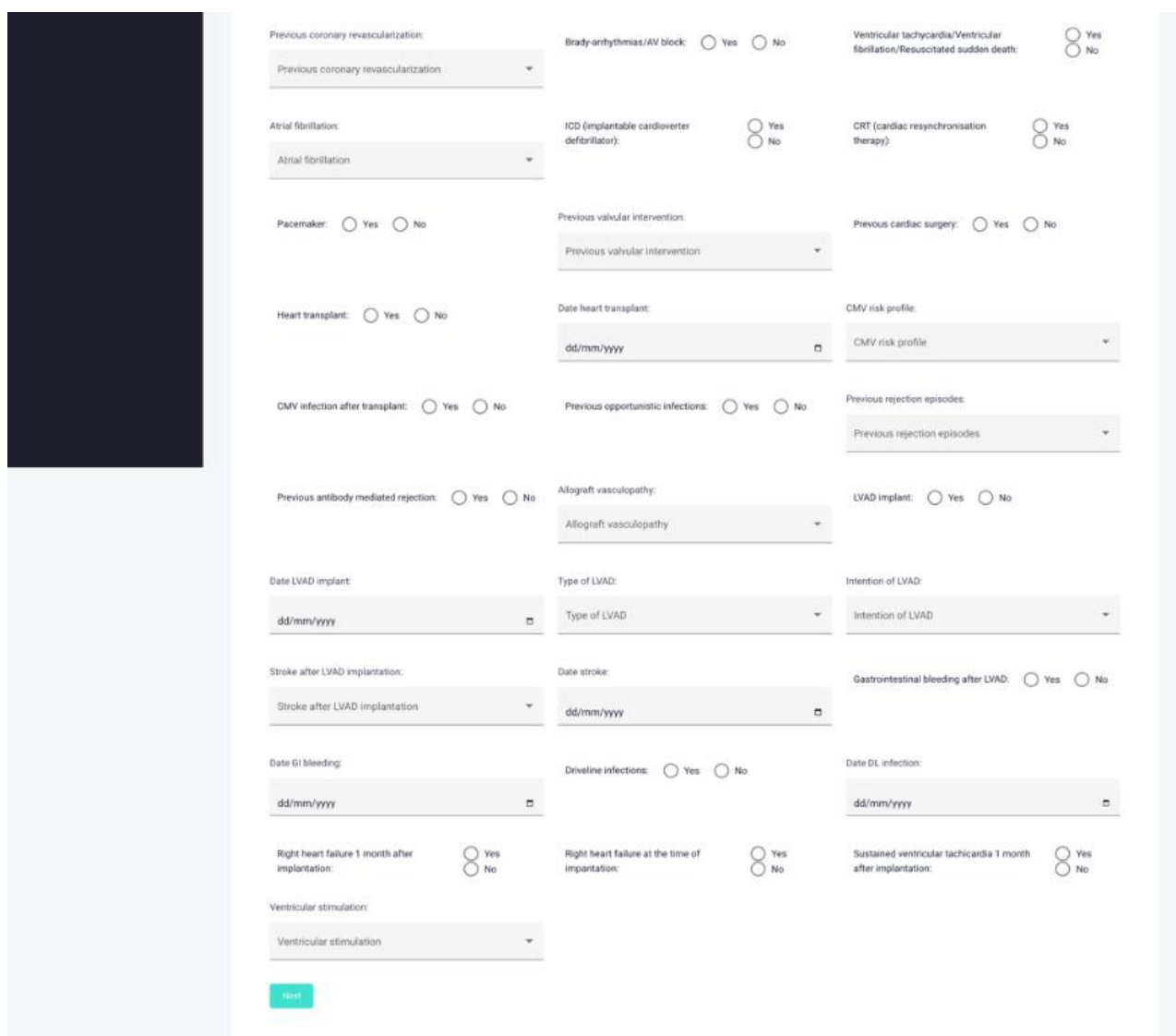
RTCL02 Patient-doctor interactions: enter and store the information during a patient visit

The CSB supports the monitoring of the visits and interaction between the clinician and the patient. Through the very first visit, the clinician creates the baseline of the patient and can update all the available fields indicated by the clinical protocol.

The screenshot displays the 'Details' page for a patient named Katerina Mylona. The interface includes a sidebar with navigation options: Patients Monitoring, Device Monitoring, Clinical Studies, Model Selection, and Notification Center. The main content area is titled 'Basic Information' and contains fields for ID (103), Patient Name (Katerina Mylona), Recruitment status (Accepted), Phone (6973643262), and Birthdate (Jul 10, 1981). Below this, there are tabs for 'Main Patient Info', 'Patient Monitoring', and 'Notifications'. The 'Patient Monitoring' tab is active, showing a 'Patient Data' section with various medical conditions and symptoms. These are organized into a grid with radio buttons for 'Yes' and 'No' or text input fields. The conditions include Arterial Hypertension, Diabetes Mellitus type 1 and 2, COPD, Renal insufficiency, Cerebrovascular disease, Peripheral artery disease, Aortic aneurysm, Venous thromboembolism, Carotid Artery Disease, Cancer, Anemia, Thyroid Disease, Benign Prostatic Hyperplasia, Smoker, Severe alcohol intake, Date HF diagnosis, Type of cardiomyopathy, NYHA class, Number of HF admissions during the previous year, Date last HF admission, and Coronary artery disease. The 'Done' button is visible at the top right of the form.

Condition	Value
ID	103
Patient Name	Katerina Mylona
Recruitment status	Accepted
Phone	6973643262
Birthdate	Jul 10, 1981
Arterial Hypertension	No
Diabetes Mellitus type 1	No
Diabetes Mellitus type 2	No
COPD	No
Renal insufficiency	No
Cerebrovascular disease (previous stroke or transient ischemic attack)	No
Peripheral artery disease	No
Aortic aneurysm	No
Venous thromboembolism / pulmonary embolism	No
Carotid Artery Disease	No
Cancer	No
Anemia	No
Thyroid Disease	
Benign Prostatic Hyperplasia	No
Smoker	Smoker
Severe alcohol intake	
Date HF diagnosis	dd/mm/yyyy
Type of cardiomyopathy	
NYHA class	
Number of HF admissions during the previous year	0-365
Date last HF admission	dd/mm/yyyy
Coronary artery disease	No

Figure 67: Patient's baseline (part 1)



The form is titled 'Patient's baseline (part 2)' and contains the following fields and sections:

- Previous coronary revascularization:** A dropdown menu.
- Bradyarrhythmias/AV block:** Radio buttons for Yes and No.
- Ventricular tachycardia/Ventricular fibrillation/Resuscitated sudden death:** Radio buttons for Yes and No.
- Atrial fibrillation:** A dropdown menu.
- ICD (implantable cardioverter defibrillator):** Radio buttons for Yes and No.
- CRT (cardiac resynchronisation therapy):** Radio buttons for Yes and No.
- Pacemaker:** Radio buttons for Yes and No.
- Previous valvular intervention:** A dropdown menu.
- Previous cardiac surgery:** Radio buttons for Yes and No.
- Heart transplant:** Radio buttons for Yes and No.
- Date heart transplant:** A date input field (dd/mm/yyyy).
- CMV risk profile:** A dropdown menu.
- CMV infection after transplant:** Radio buttons for Yes and No.
- Previous opportunistic infections:** Radio buttons for Yes and No.
- Previous rejection episodes:** A dropdown menu.
- Previous antibody mediated rejection:** Radio buttons for Yes and No.
- Allograft vasculopathy:** A dropdown menu.
- LVAD implant:** Radio buttons for Yes and No.
- Date LVAD implant:** A date input field (dd/mm/yyyy).
- Type of LVAD:** A dropdown menu.
- Intention of LVAD:** A dropdown menu.
- Stroke after LVAD implantation:** A dropdown menu.
- Date stroke:** A date input field (dd/mm/yyyy).
- Gastrointestinal bleeding after LVAD:** Radio buttons for Yes and No.
- Date GI bleeding:** A date input field (dd/mm/yyyy).
- Driveline infections:** Radio buttons for Yes and No.
- Date DL infection:** A date input field (dd/mm/yyyy).
- Right heart failure 1 month after implantation:** Radio buttons for Yes and No.
- Right heart failure at the time of implantation:** Radio buttons for Yes and No.
- Sustained ventricular tachycardia 1 month after implantation:** Radio buttons for Yes and No.
- Ventricular stimulation:** A dropdown menu.
- Next:** A green button.

Figure 68: Patient's baseline (part 2)

The user/clinician can navigate to an overview all patient visits (Tab: Patients Monitoring, sub-tab: 'Visit' and can also add the event of a new visit with additional details.



Patients Monitoring

Device Monitoring
Clinical Studies
Model Selection

Notification center

2022/11/16 16:41
Patient: Katerina Mylona
Severity: critical

2022/11/16 16:25
Patient: Katerina Mylona
Severity: low

2022/12/09 13:51
Patient: Katerina Mylona
Severity: low

2022/12/12 15:19
Patient: Katerina Mylona

Details

Basic Information

ID: 103
Patient: Katerina Milona
Recruitment status: Accepted

Phone: 6973643262
Birthdate: Jul 10, 1981

Basic Patient Info **Patient Monitoring** **Notifications**

Overview Visits Medication Personalized Analytics Images ClinicalData ECG Labs Echocardiography 6MWT VAD CardioPulmonary Caregiver RWD Sensors OtherData

Add Visit

	Date	Type of Visit	
☐	Mon, 19 Dec 2022 14:27	call	Delete
☐	Tue, 11 Dec 2022 14:27	live	Delete
☐	Mon, 10 Dec 2022 14:26	call	Delete

Figure 69: Visits history

Basic Information

ID: 103
Patient: Katerina Milona
Recruitment status: Accepted

Phone: 6973643262
Birthdate: Jul 10, 1981

Basic Patient Info **Patient Monitoring** **Notifications**

Overview Visits Medication Personalized Analytics Images ClinicalData ECG Labs Echocardiography 6MWT VAD CardioPulmonary Caregiver RWD Sensors OtherData

Add Visit

	Date	Type of Visit	
	Mon, 19 Dec 2022 14:21	live	Done
☐	Mon, 19 Dec 2022 14:27	call	Delete
☐	Tue, 11 Dec 2022 14:27	live	Delete
☐	Mon, 10 Dec 2022 14:26	call	Delete

Figure 70: New visit creation

RTCL03 Alarms: Important patients' alarms and detected decompensations

At the left vertical side of the dashboard, there is the info block 'Notification Center' that displays several notifications per patient, grouped by the criticality of the notification. It is noted that the notifications are created by the DSS component and the block presents the notifications that have status 'unread'.

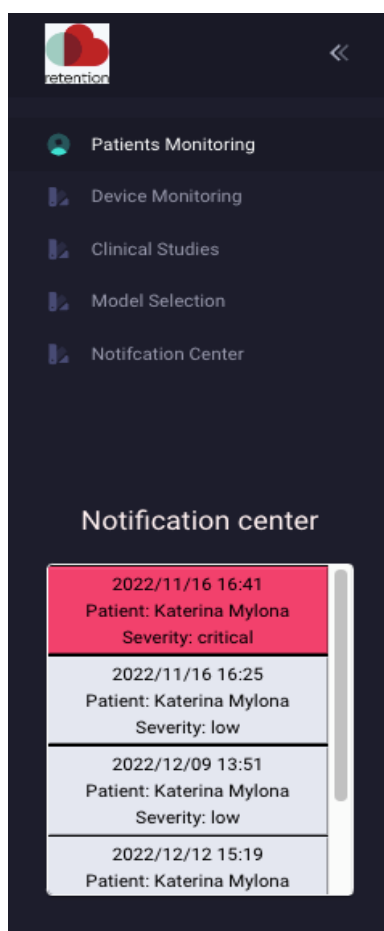
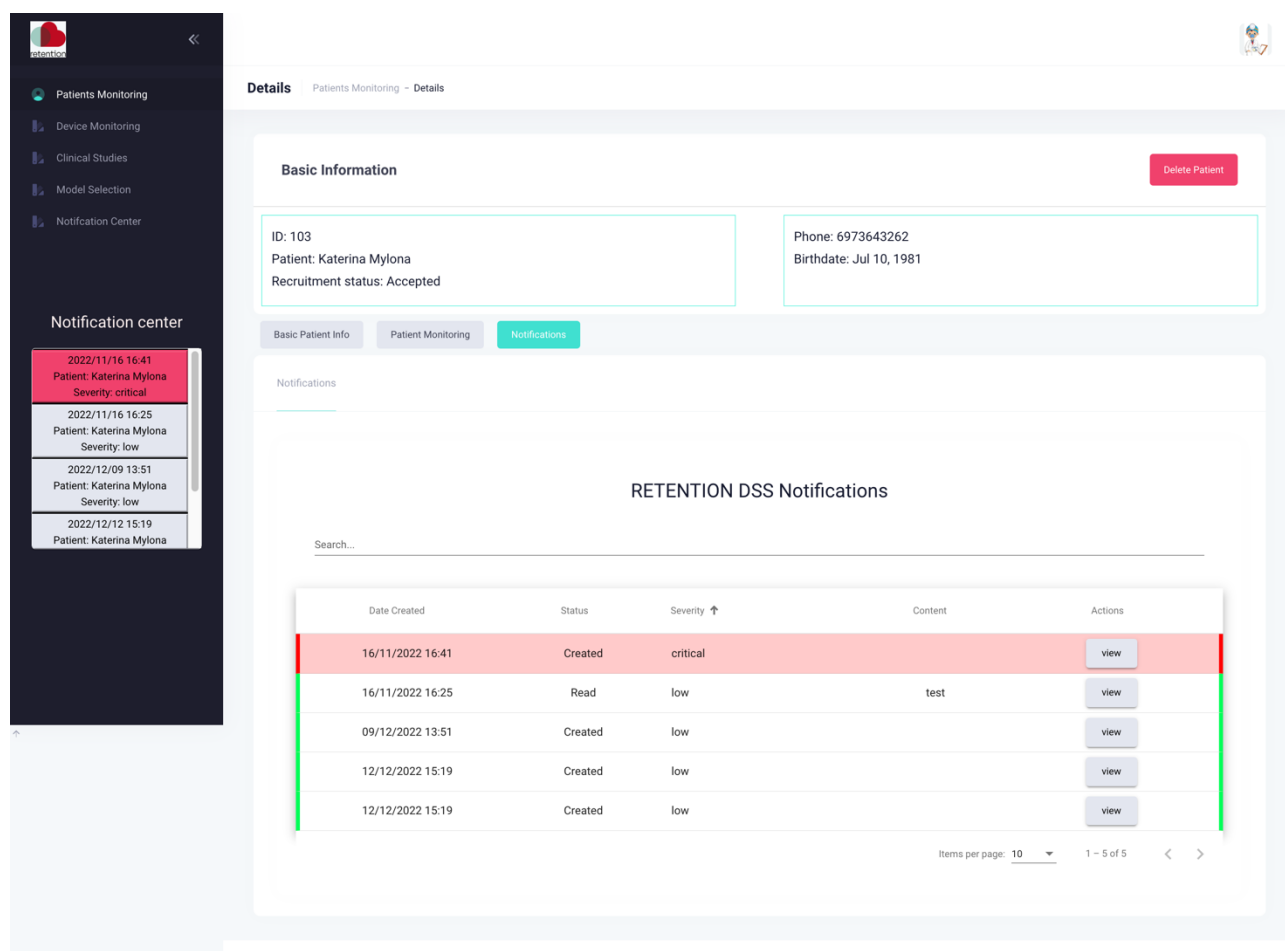


Figure 71: CSB Notification Centre

By clicking a specific notification from the block, the user navigates to the profile of the patient and the table of his notifications, in specific. The clinician can sort the available notifications (by status, criticality etc.), can have a short overview of all notifications and a can view the one she needs her action. After viewing a notification, the status of the notification is changed to 'read'.



Notification center

- 2022/11/16 16:41
Patient: Katerina Mylona
Severity: critical
- 2022/11/16 16:25
Patient: Katerina Mylona
Severity: low
- 2022/12/09 13:51
Patient: Katerina Mylona
Severity: low
- 2022/12/12 15:19
Patient: Katerina Mylona

Details Patients Monitoring - Details

Basic Information [Delete Patient](#)

ID: 103
Patient: Katerina Mylona
Recruitment status: Accepted

Phone: 6973643262
Birthdate: Jul 10, 1981

Basic Patient Info Patient Monitoring **Notifications**

Notifications

RETENTION DSS Notifications

Search...

Date Created	Status	Severity ↑	Content	Actions
16/11/2022 16:41	Created	critical		view
16/11/2022 16:25	Read	low	test	view
09/12/2022 13:51	Created	low		view
12/12/2022 15:19	Created	low		view
12/12/2022 15:19	Created	low		view

Items per page: 10 1 - 5 of 5

Figure 72: Patient's notification screen

RTCL04 Event record: every event occurring to the patient

The CSB provides the functionality to track the events related to a specific patient and also provides the form to create a new event.

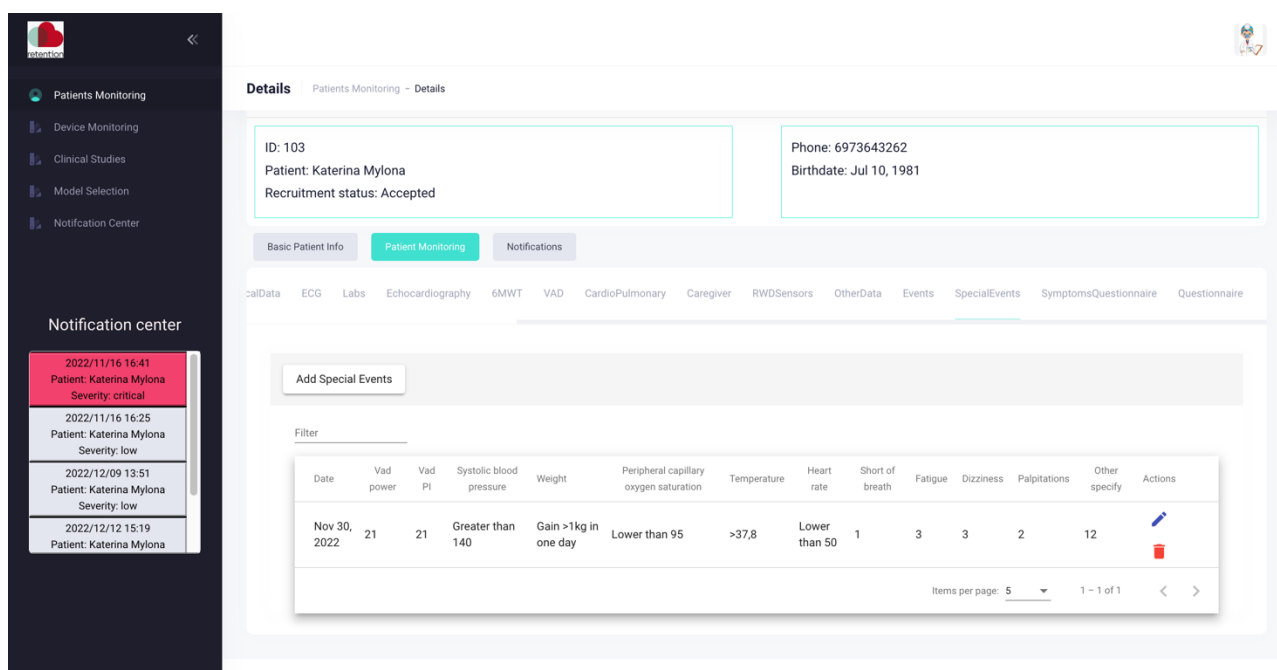


Figure 73: Patient's events history

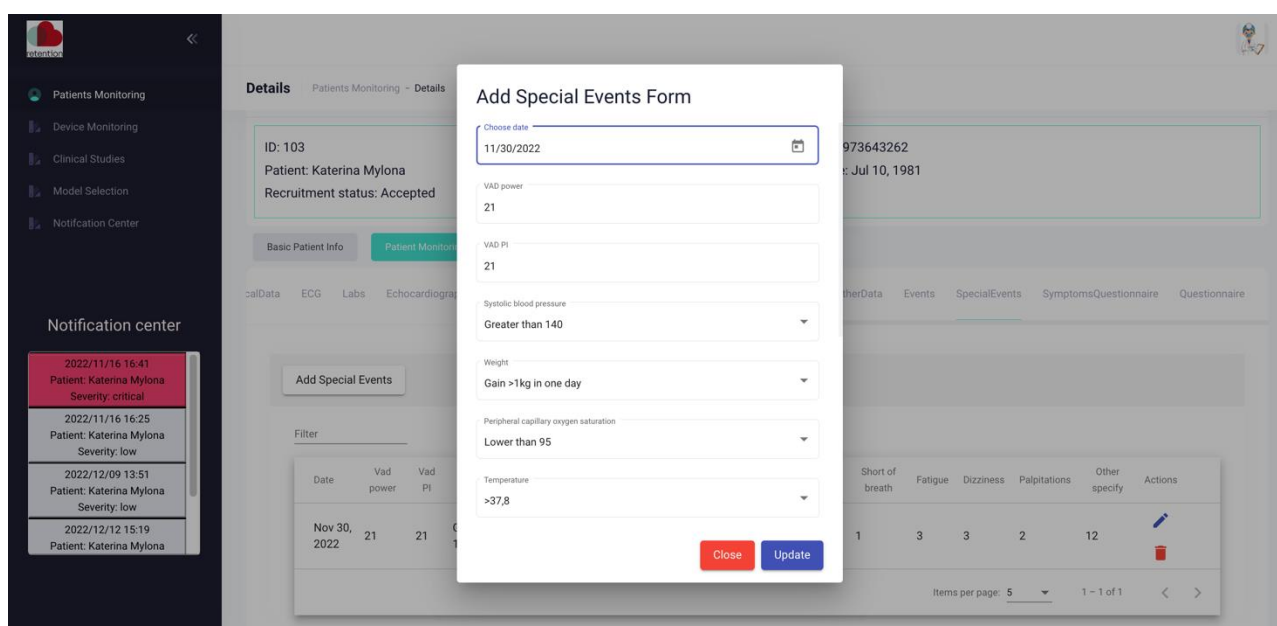


Figure 74: Creation of a new event for a patient

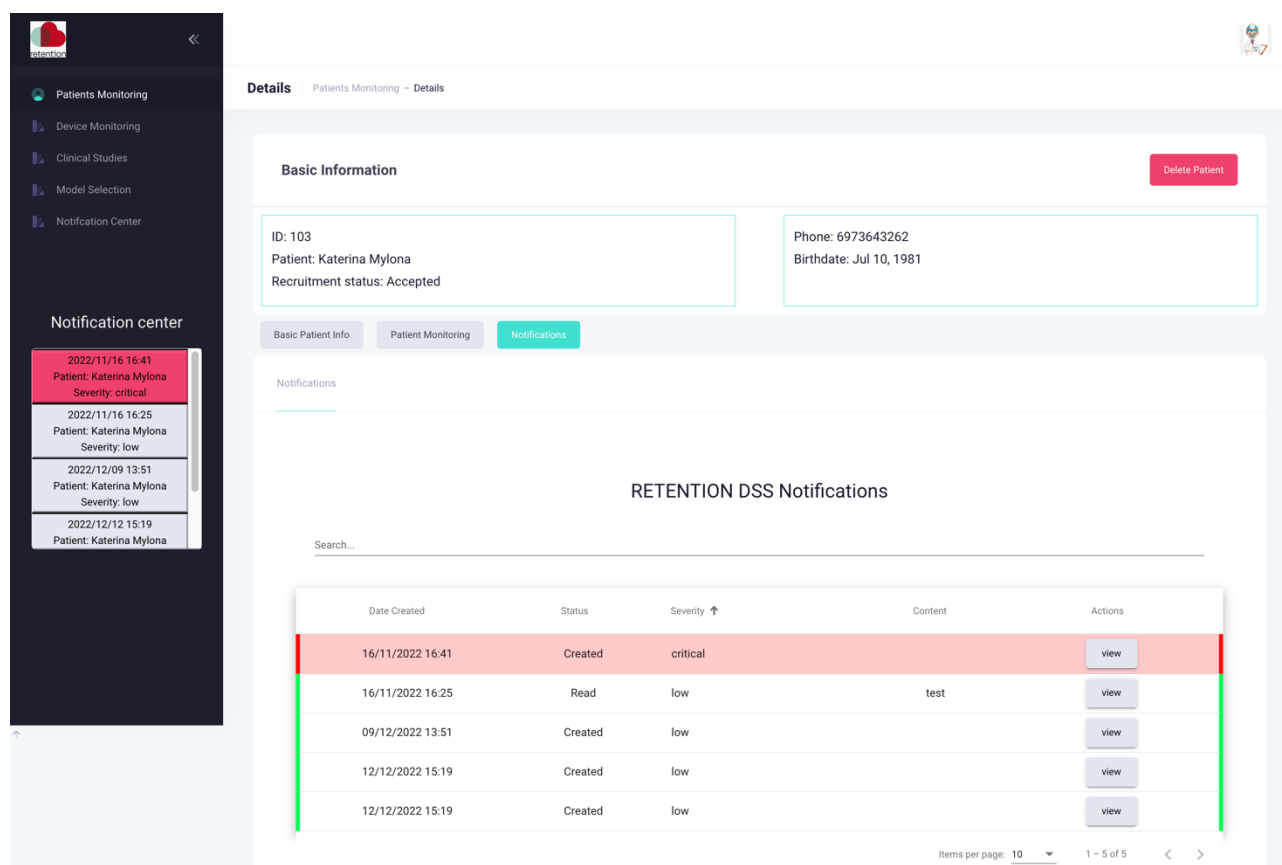


Figure 75: Automatic messages design

4.3.2 Risk assessment generated by AI

RTBD01 Risk assessment generated by AI

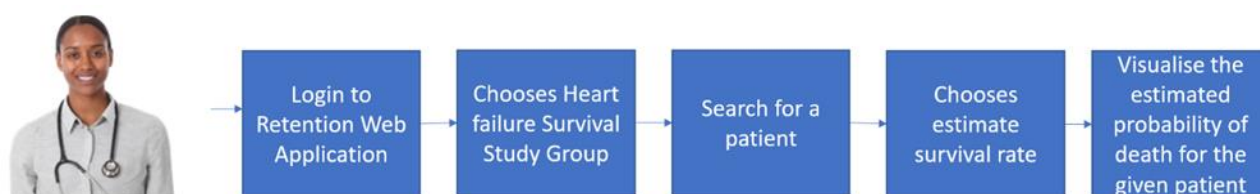


Figure 76: "Heart Failure" Scenario workflow

The above flow is realised through the vertical menu 'Clinical Studies'. Figure 77 shows what the clinician sees when choosing the "Clinical Studies" menu. In the upper part, there is a "Clinical study" dropdown that allows the doctor to select the scenario he or she would like to investigate. Once this selection is made, a list of patients will be updated and for each patient, in these current developments phase, there is the option to choose "Estimate". Once this button is clicked, the gauge item below will indicate the certainty that the patients will die within the next 250 days; this is predicted by the AI system according to the specific patient's variables displayed in the lower part of the form.

After the selection or search for the specific patient and clicking the button 'Estimate', the estimated prediction for the patient will be shown, graphically represented in a colour bar, along with the patient's data

from the FHIR database, shown in a table. In this case of Ahmed Juliana, the follow-up time is considered as of 285 days with measurements from the Report Note Date: Sept'22. The threshold for deciding on the probability percentage would be around 50%, meaning patient is estimated to die if probability is above 45% (high risk) and estimated to survive if probability is below 45%, with medium (15-45%) or low risk (0-15%) of Heart Failure. For values in the range 45-55%, it is hard to make a decision on the probability but is considered here as high risk. For this example, based on its data measurements, the estimation shows: 75% risk of death in the following 250 days, meaning that after Sept'22 high risk of death with a probability of 75%.

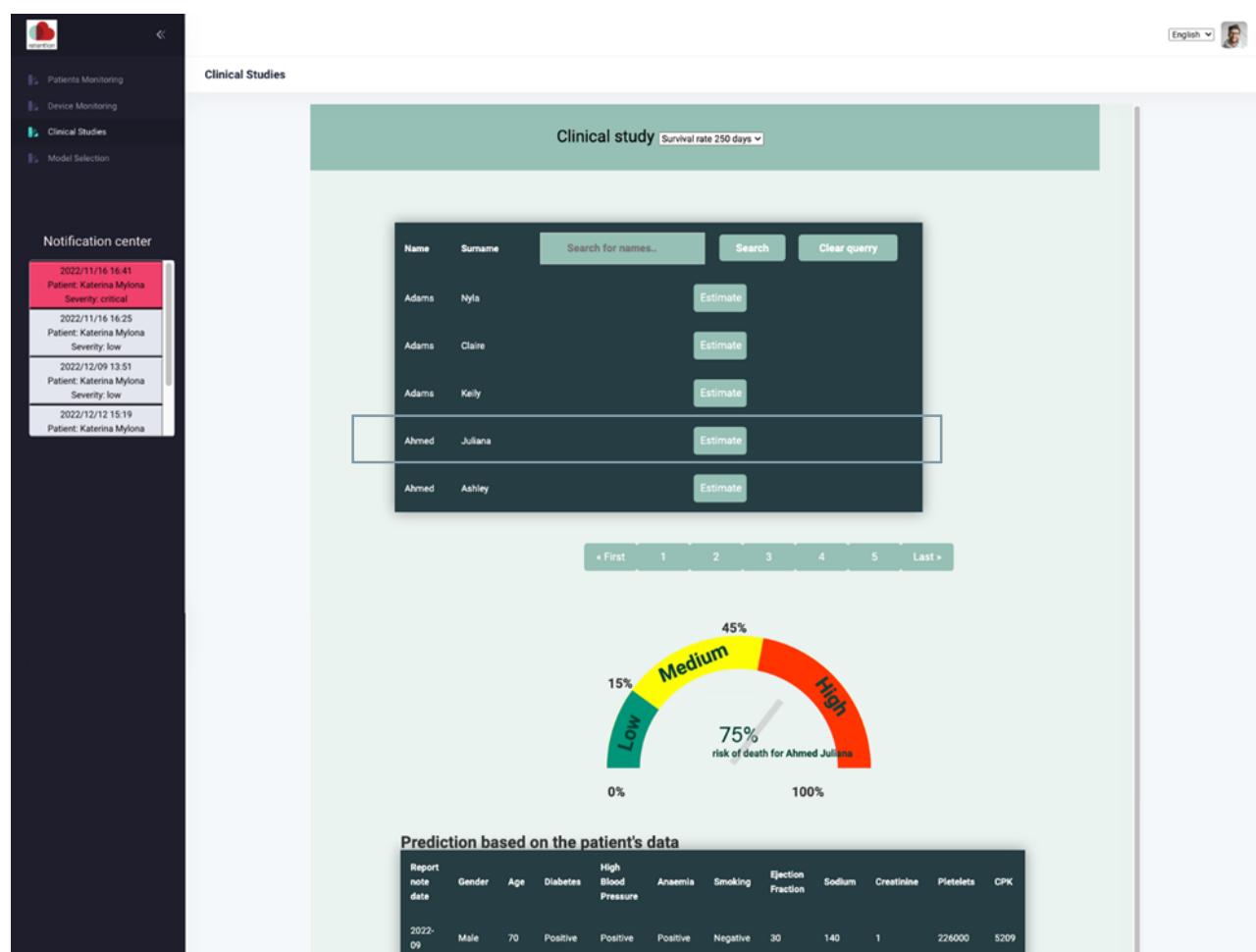


Figure 77: Clinical studies screen.

4.3.3 Technical use cases

Under the heading technical use cases, D3.1 specified auxiliary functionality, essential for providing the core functionality demonstrated above to the intended users of the CSB Dashboard, facilitating the collection of data from a patient's home and wearable devices etc.

TS01 Dashboard end-user moderated registration

At the landing page of the CSB, the user can login (if an account already exists) or sign up and request access. After completing all required info, the user's registration request has the status 'unverified' and an Admin should review his/her request and grant access by altering the status of the user to 'active'.

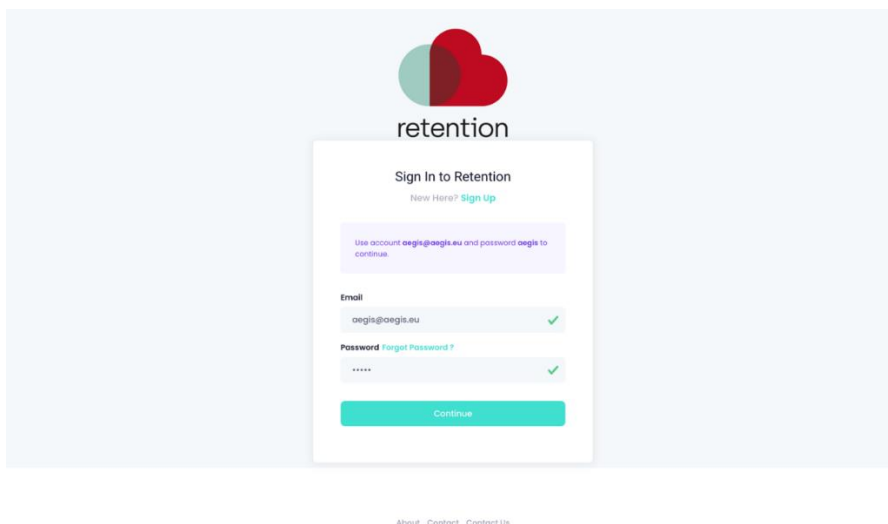


Figure 78: Login and/or Sign Up page of CSB

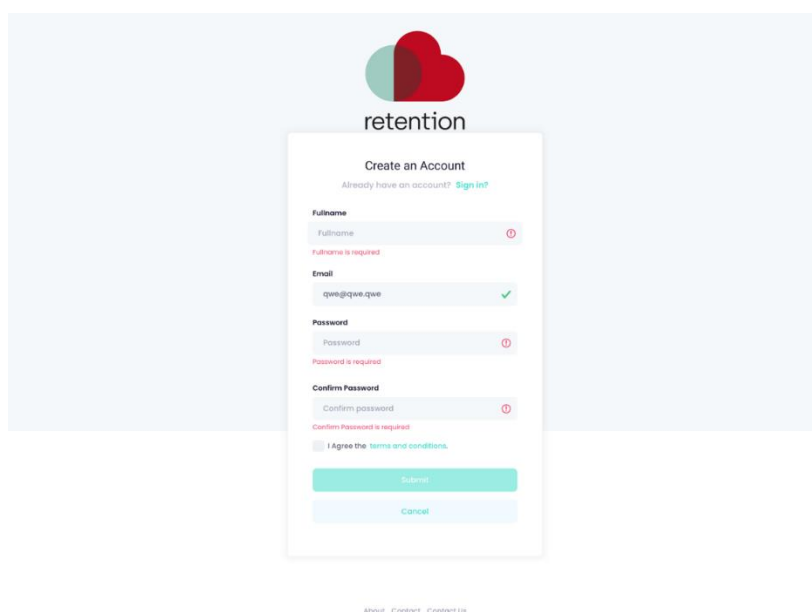


Figure 79: Sign Up page for a new user

A **CSB admin** (the user role is admin) has a dedicated menu 'End-user management' for reviewing all pending access requests.

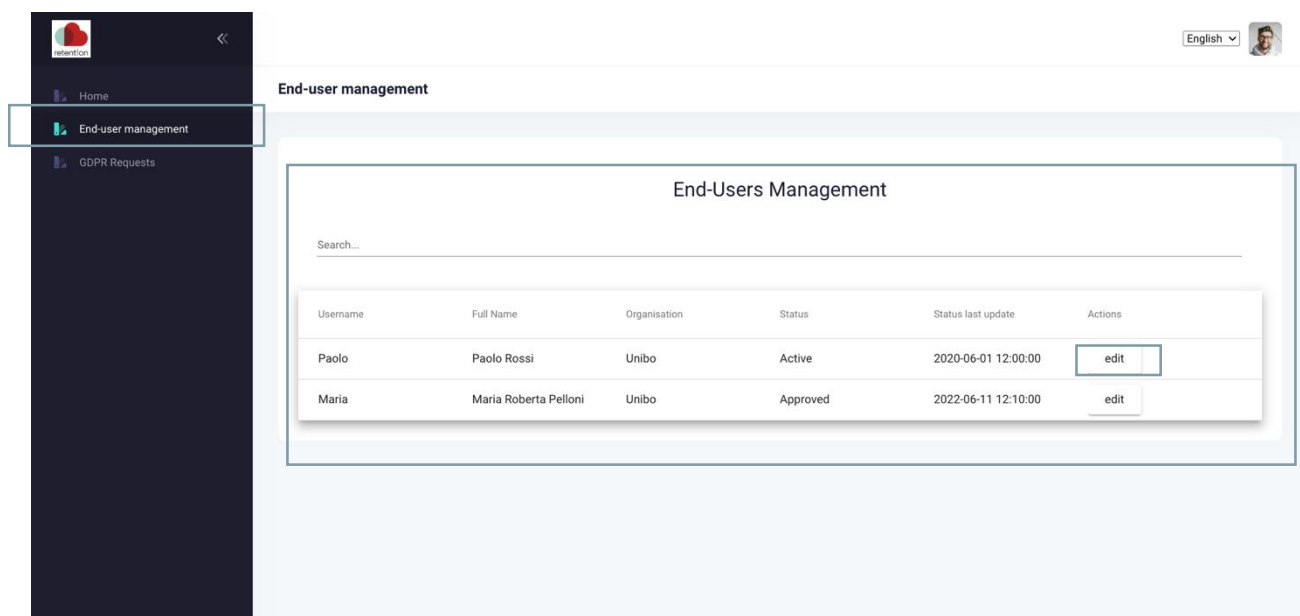


Figure 80: End-user management (Admin role)

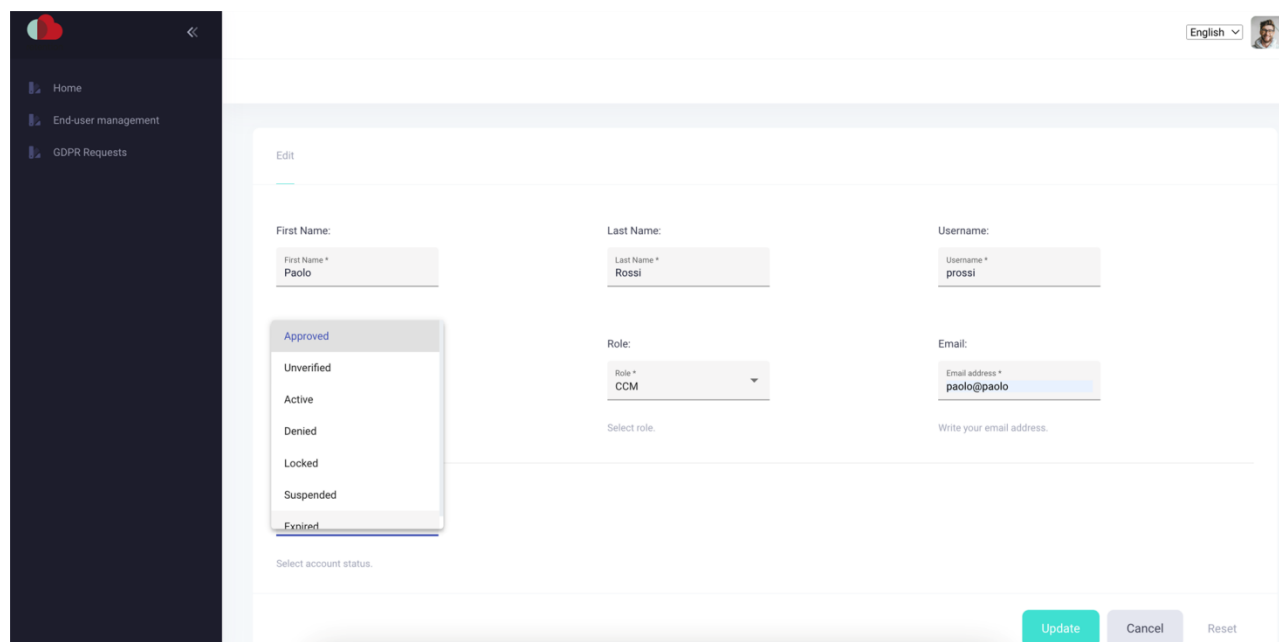


Figure 81: Grant access after reviewing (Admin role)

RTS02 Dashboard End-user login

As per the D3.1 requirement, only authenticated users have access to CSB Dashboard functionality, and what functionality they have access to is dictated by their assigned role.

For example, the technician's main menu allows only the 'Device Monitoring' and the configuration of the monitoring devices (Figure 82). An overview of Clinician's dashboard is presented in Figure 24: CSB Frontpage for the Role 'Clinician' and for the role 'ADMIN' in Figure 83

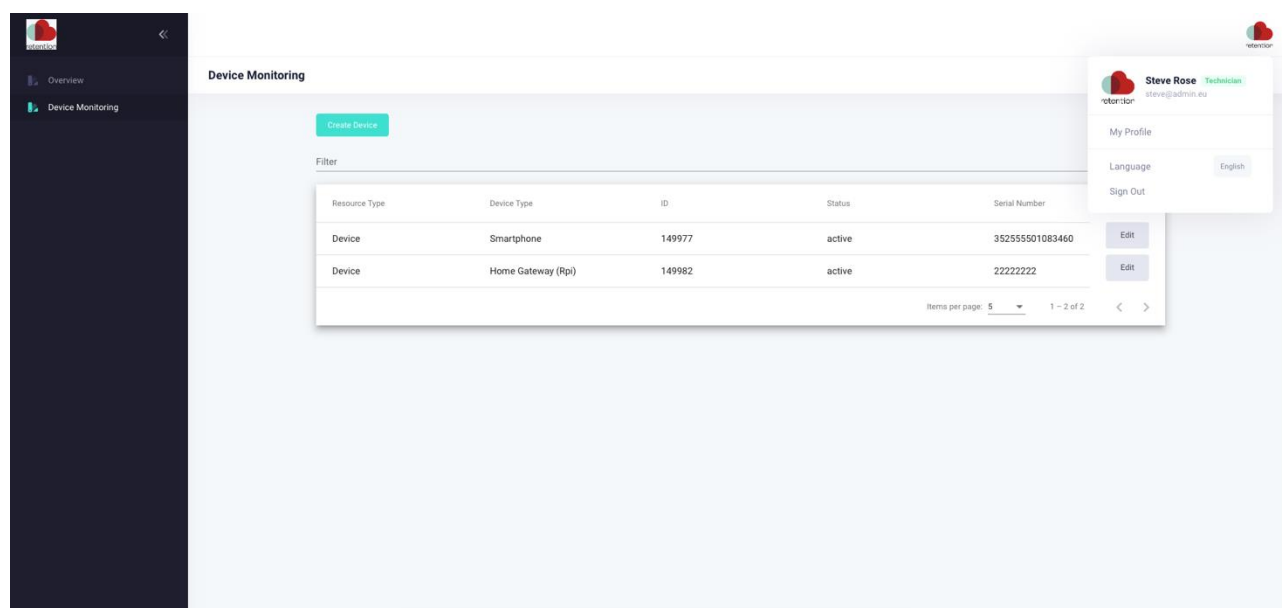


Figure 82: Technician's Dashboard

RTS03 Manage end-user account information

Regardless the role, after logging in, users can manage their profile and change their password.

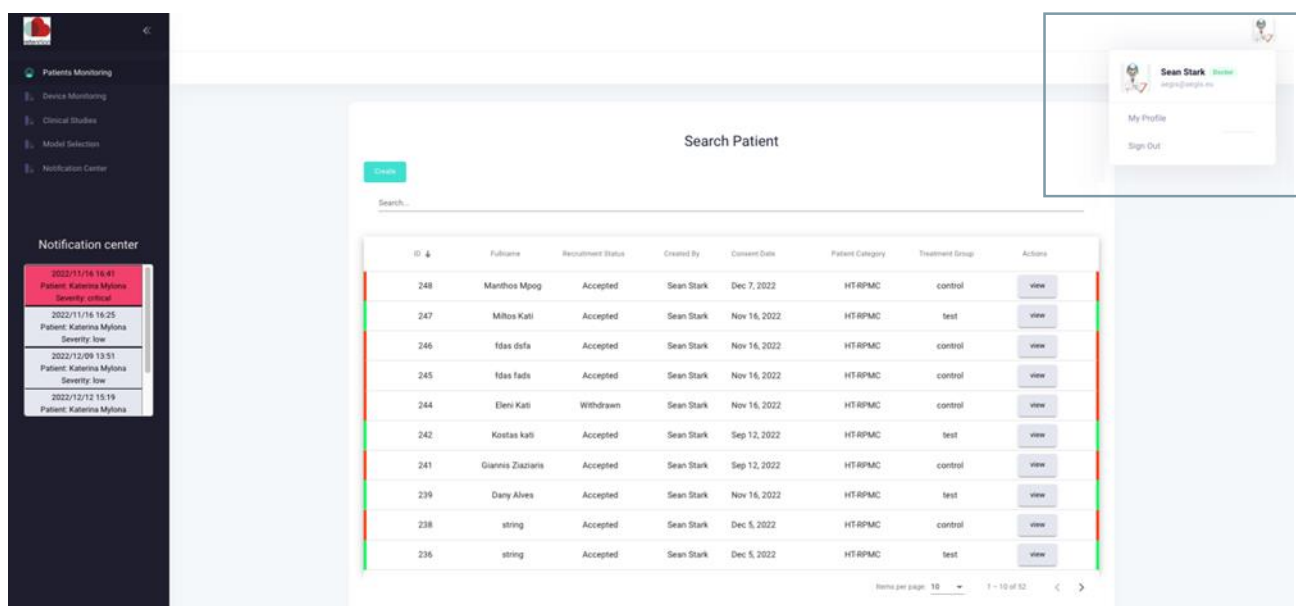


Figure 83: Manage end-user account information

RTS04 Associating a device ID to a patient

A clinician can assign a device to a specific patient, after entering the patient's profile (Patient monitoring, basic Patient info, Devices). In this specific screen, there is a table with all patient's associated devices. The clinician can assign a new device (button 'Assign Device').

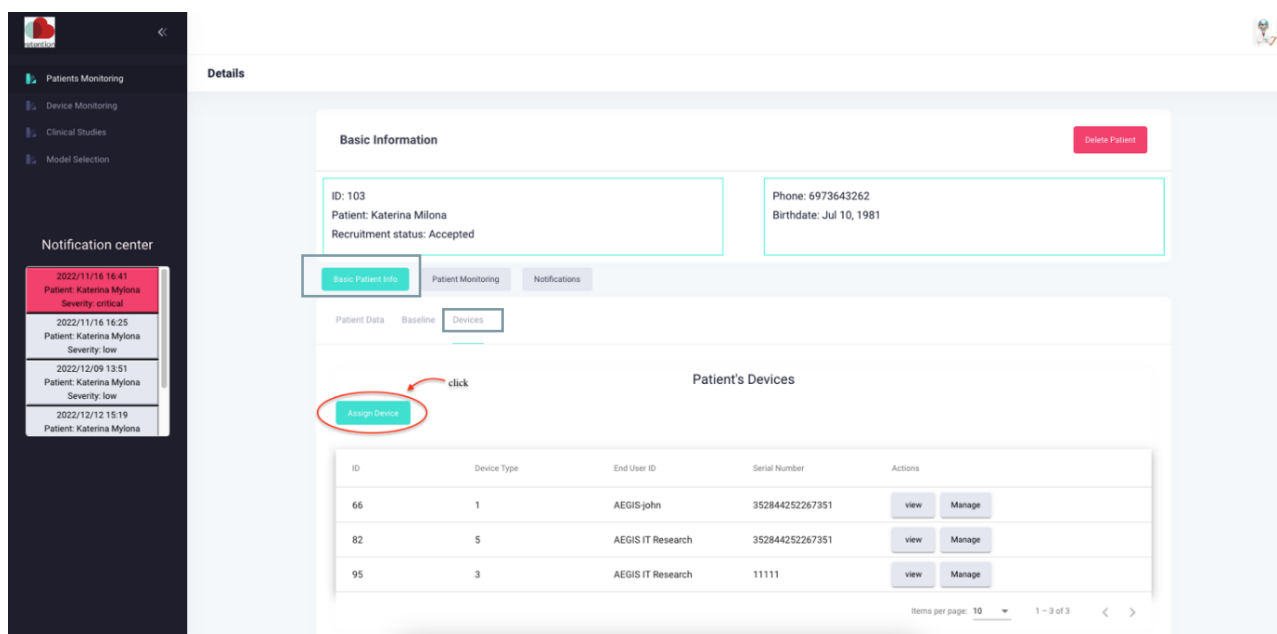


Figure 84: Patient's assigned devices

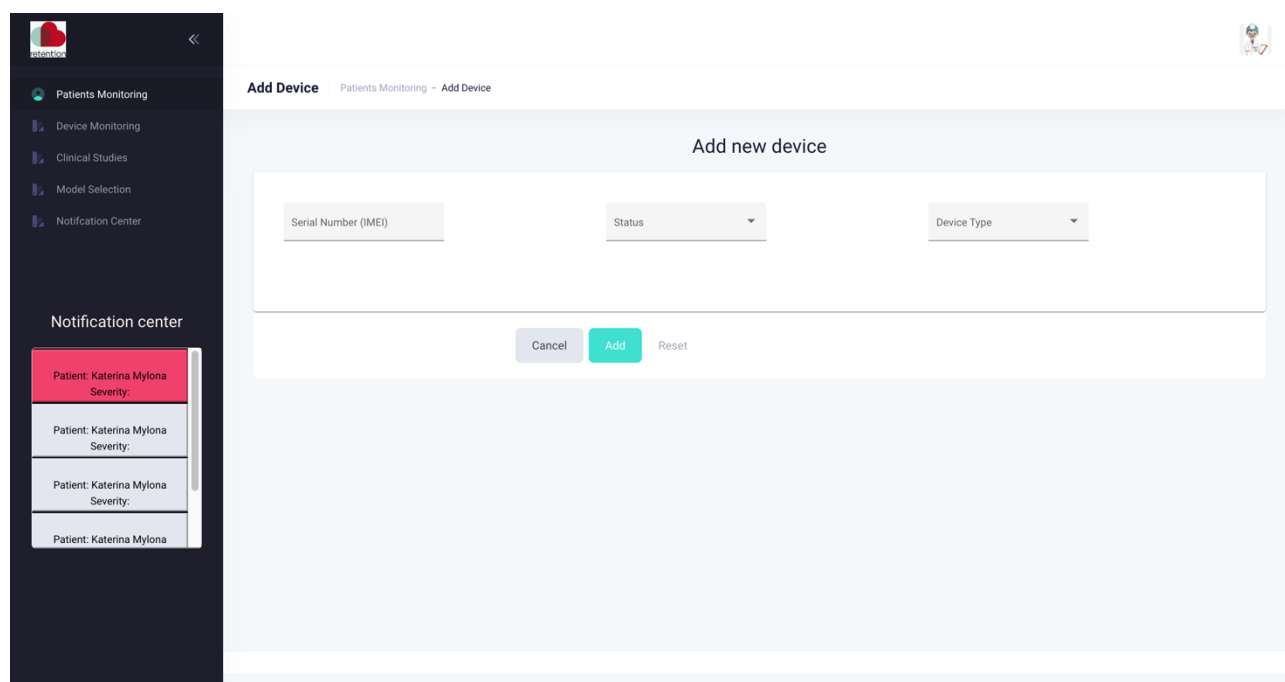


Figure 85: Assign a new device to patient

4.4 Data flows

To better present the flow of actions that a clinician will perform, and to grasp the implementation effort for this GUI form, we will present several sequence diagrams for specific type of actions that take place within CSB.

In Figure 86, the actions that one doctor must perform in order to introduce patient's information into the RETENTION system are presented. This step is crucial because using this data, the data scientist will train models that will be delivered back to the clinician.

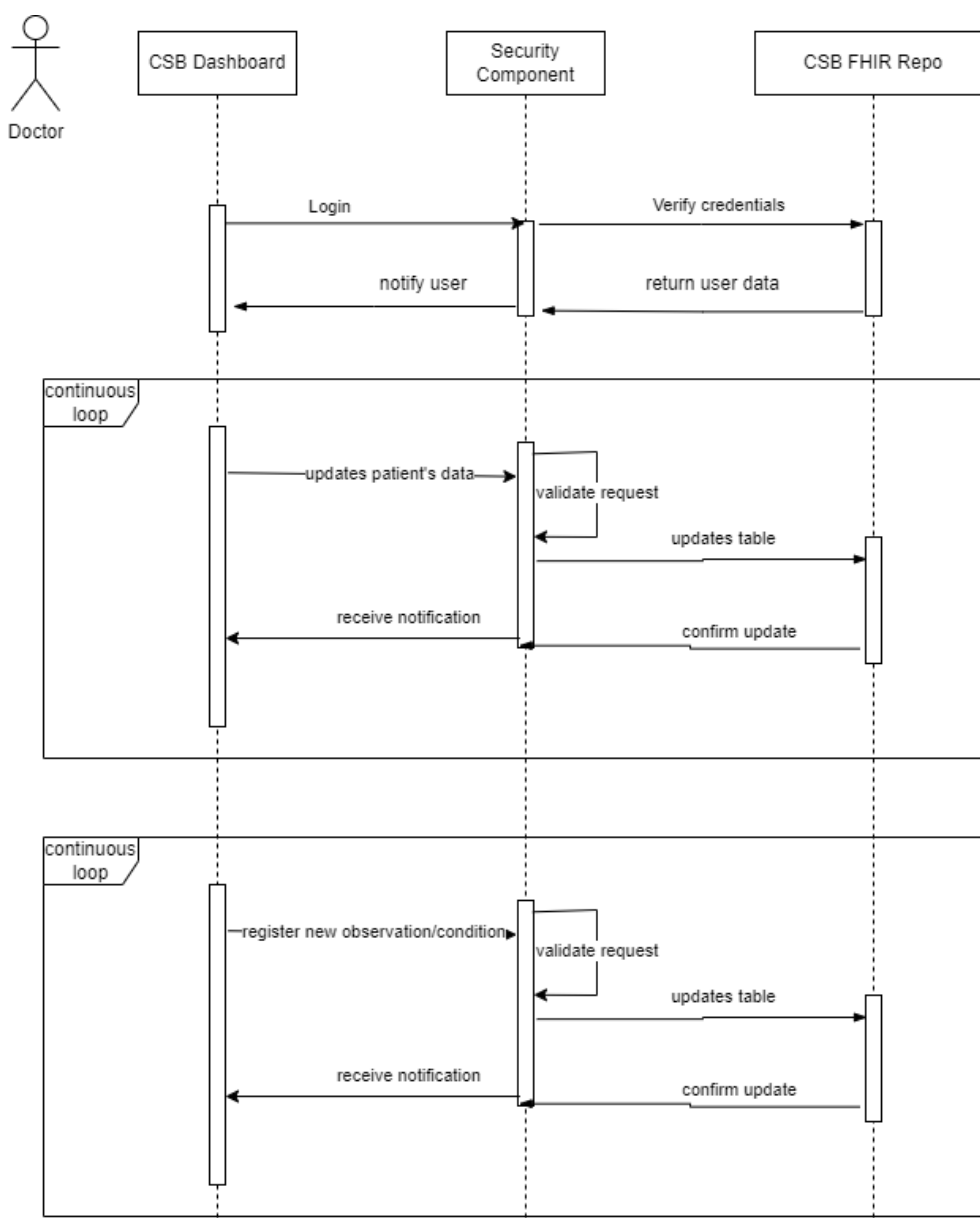


Figure 86: Clinician actions regarding patient data

In Figure 87, presumably after the data scientist has created the models, we show what are the steps that the doctor must perform in order to get insights about his or her patient evolution, insights supported by AI algorithms.

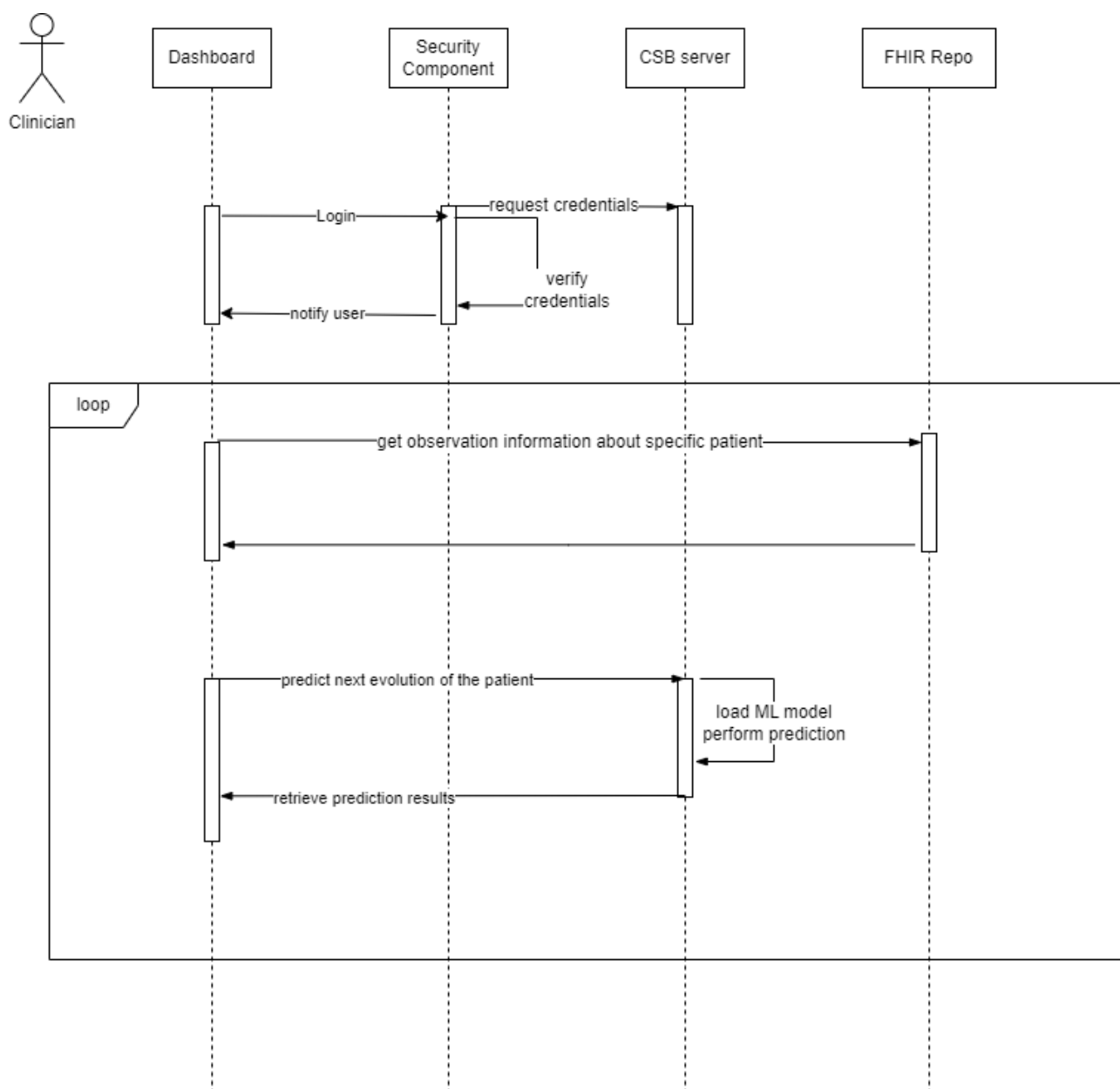


Figure 87: Clinician Prediction Flow

4.5 Configuration details

Each Clinical Site Backend will have an identical layout and will serve the same functionalities and user journeys presented in section 4.1. From technical point of view, each CSB dashboard should be configured accordingly in order to link to each hospital data CSB repos and components, configured by a YAML file.



5 RETENTION Global Insights Cloud Dashboard & Visualisations

This section presents the Global Insight Cloud dashboard and the implementation of the server side where the models are created and managed according to the architecture described in D5.1

5.1 Description and objectives

The objective of the GIC dashboard is to accommodate different views suited for the users expected to interact with the GIC, including the following categories: (i) clinical experts and data scientists, (ii) healthcare policy makers, and (iii) IT/administrative staff. Through the available dashboards, the user can access functions that provide

- model view and specification,
- analytics execution and monitoring of the federated big data analytics,
- disease and policy insights and other reporting-focused views related to the metanalysis of the available data,
- a view dedicated to the connection status of the underlying RETENTION CSB instances (e.g., to monitor if each of those is online/offline, the last time data was received, possible alerts),
- configuration and administrative panel,
- a view coming from the Continuous Security and Privacy Assurance tool, providing a real-time, global view of the security and privacy posture of the GIC itself, but also of the reported states received by the underlying CSB instances.

The current version of the GIC focuses mainly on the data scientists and clinical experts and provides the means for creating, deploying and sharing models in a scalable and easy to explain fashion ('RTS06 Managing Public health policy decision-making models'). Also, it supports the following technical staff use case 'RTS01 Dashboard End-user moderated registration'.

5.2 Dashboard tools and libraries

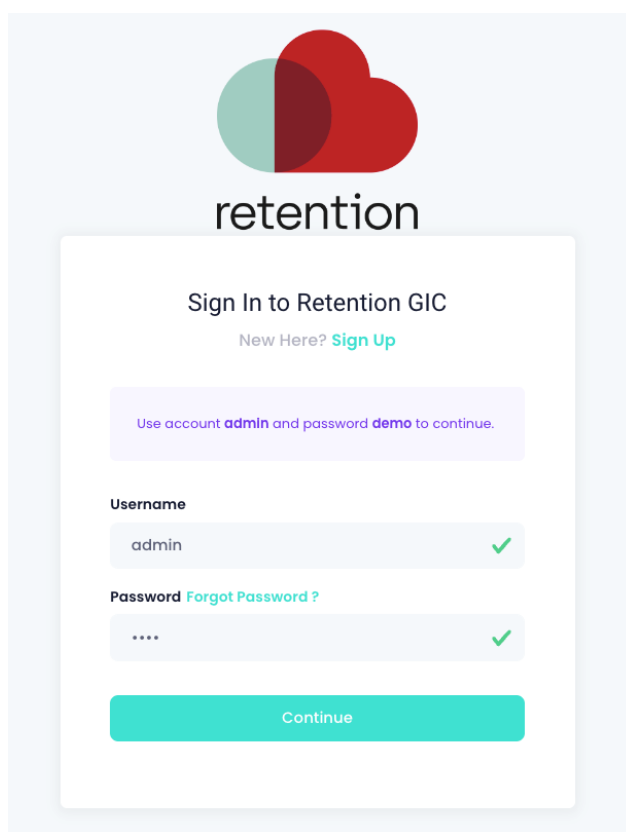
The Global Insights Cloud Dashboard is built based on AVT, using the same technologies of CSB, described in Section 4.2. It incorporates two third-party open-source tools:

- MLflow (<https://mlflow.org/>), an open-source platform designed to manage the ML operations, including experimentation, reproducibility, deployment, and maintaining a central model registry.
- Jupyter Notebook (<https://jupyter.org/>), an open-source web application for creating and sharing documents containing live code, equations, visualizations and narrative text.

5.3 User journeys

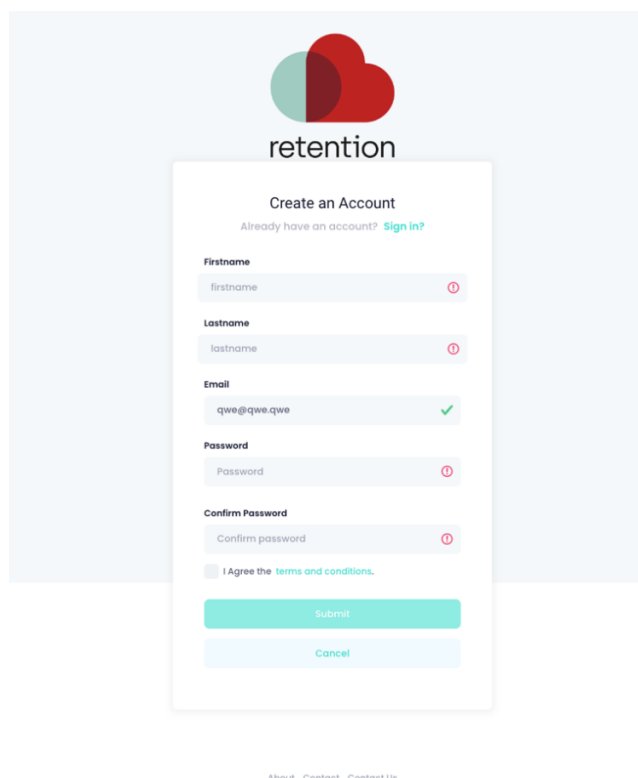
RTS06 Managing Public health policy decision-making models

The managing of the Public health policy decision-making models is done within the provided Jupyter Notebooks. The data scientist is able to elaborate various scenarios, writing code that will process the FHIR data in order to create the desired models. Only a registered and approved user can access the available tools. The login and registration process is similar to the process followed in CSB.



The image shows the 'Sign In to Retention GIC' screen. At the top is the Retention logo (a stylized cloud with green and red segments) and the word 'retention'. Below this is a white card with the title 'Sign In to Retention GIC' and a link 'New Here? Sign Up'. A purple box contains the text 'Use account **admin** and password **demo** to continue.' Below this are two input fields: 'Username' with the value 'admin' and a green checkmark, and 'Password' with masked characters '****' and a green checkmark. A link 'Forgot Password?' is next to the password field. At the bottom is a large teal 'Continue' button.

Figure 88: GIC login / sign up screen



The image shows the 'Create an Account' screen. At the top is the Retention logo and the word 'retention'. Below this is a white card with the title 'Create an Account' and a link 'Already have an account? Sign in?'. The form has five input fields: 'Firstname' (placeholder 'firstname'), 'Lastname' (placeholder 'lastname'), 'Email' (placeholder 'qwe@qwe.qwe' with a green checkmark), 'Password' (placeholder 'Password'), and 'Confirm Password' (placeholder 'Confirm password'). Below these fields is a checkbox 'I Agree the terms and conditions.' and two buttons: a teal 'Submit' button and a light blue 'Cancel' button. At the bottom of the screen, there are links 'About', 'Contact', and 'Contact Us'.

Figure 89: User registration form in GIC

After a successful login, the data scientist can access the available options regarding the ML Flow and the Jupyter Notebooks

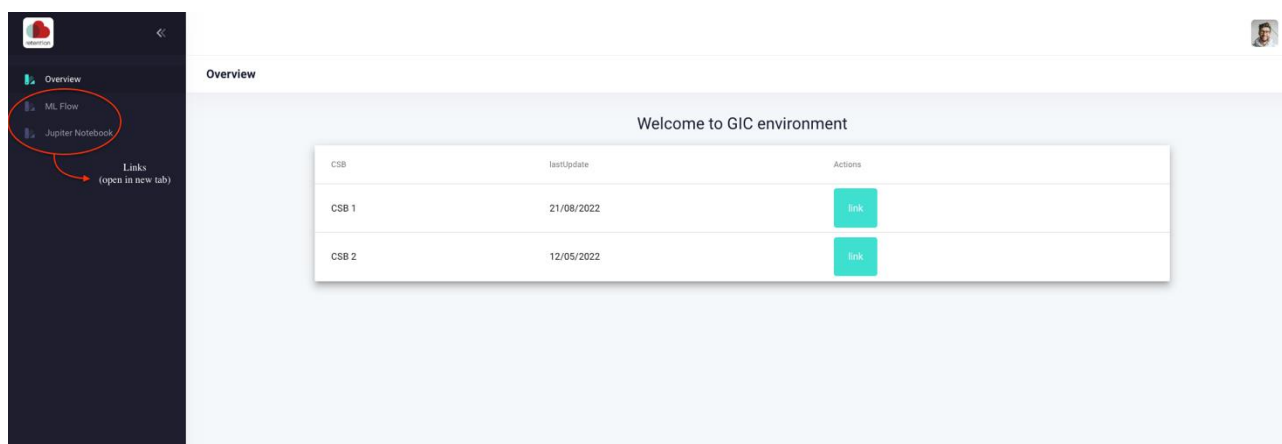


Figure 90: GIC data scientist's tools

5.4 Data flows

By means of sequence diagrams, we will depict the flow that the Data Scientist will follow in order to create and store a model along with additional meta-data, which will be described in what follows.

In Figure 91, we present the model management flow in our current vision. This version of the deliverable will be updated, so it's very likely that the diagrams will be updated if new suggestions or recommendations will appear.

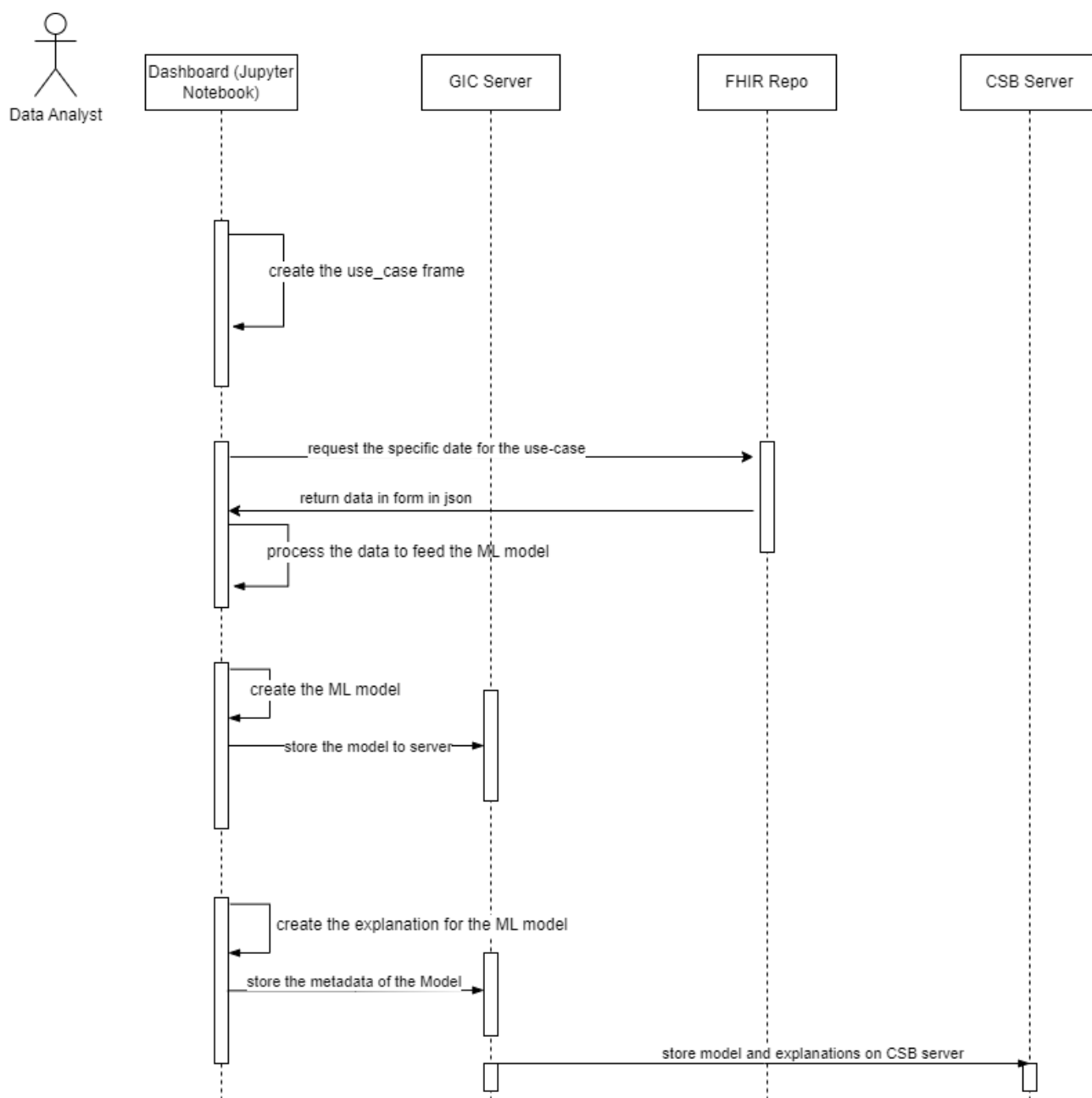


Figure 91: Model management flow



Deliverable 6.1 - RETENTION Interfacing, Device Federation and Visualisation components v1-
v1.0_28/02/2023
Dissemination Level: Public

5.5 Using MLFlow

MLFlow was presented in D5.1. Using MLFlow, the data scientist can store the model once the training has been completed as presented in Figure 93 and Figure 94.

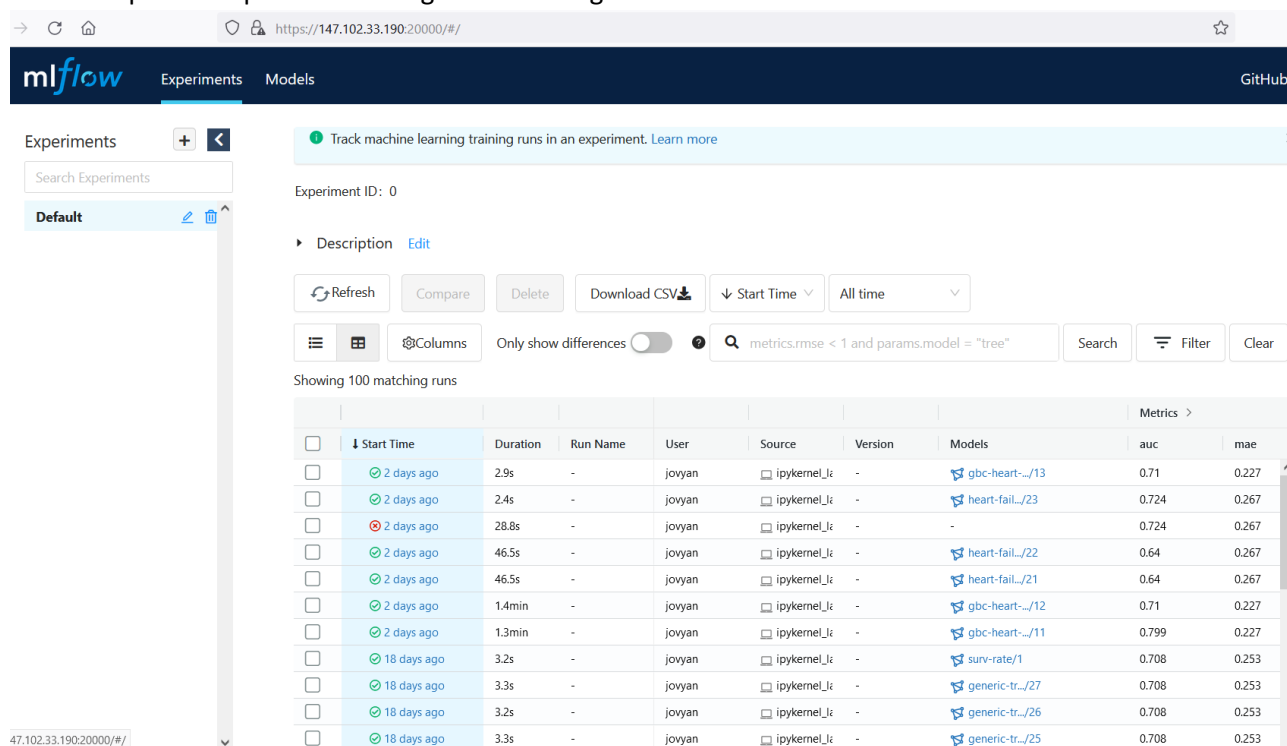


Figure 93: ML Flow Experiments

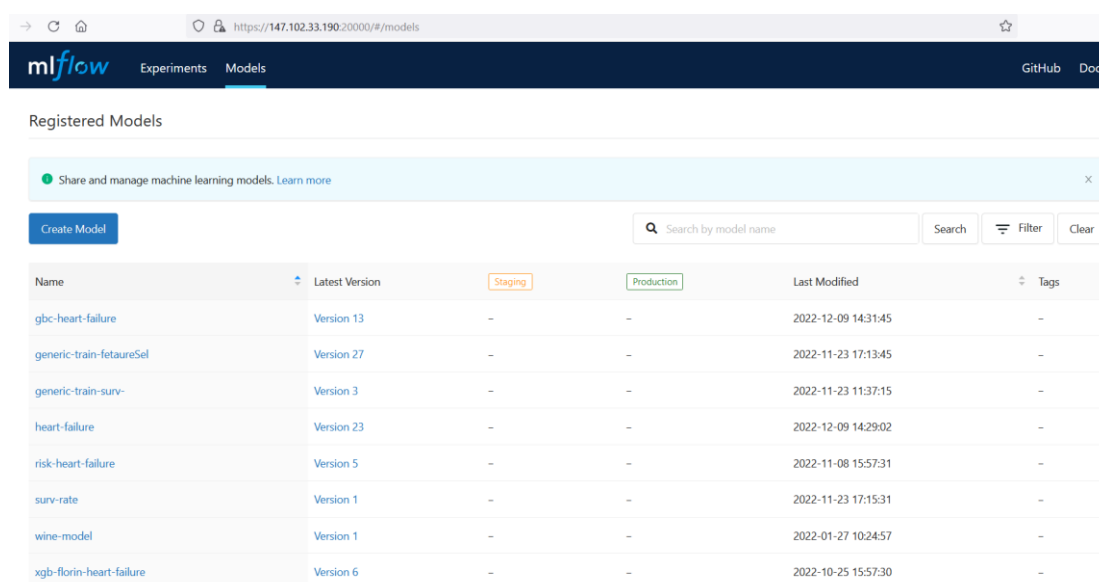
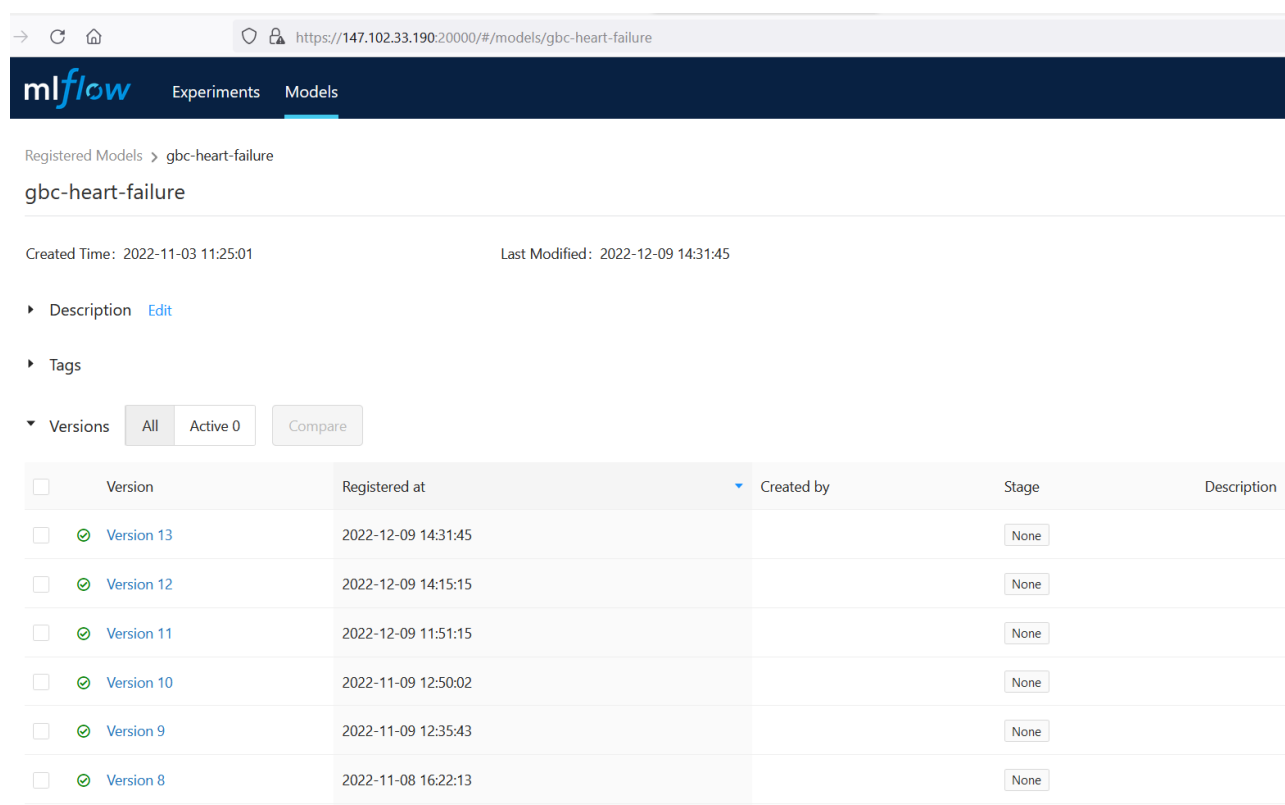


Figure 94: List of models, ML Flow Models

With help of this GUI, the data scientist can see when the model was created, how long the training took, and the model's metrics.



The screenshot shows the mlflow web interface. The browser address bar displays the URL: https://147.102.33.190:20000/#/models/gbc-heart-failure. The mlflow logo is in the top left, with 'Experiments' and 'Models' tabs. The page title is 'gbc-heart-failure'. Below the title, it shows 'Created Time: 2022-11-03 11:25:01' and 'Last Modified: 2022-12-09 14:31:45'. There are sections for 'Description' (with an 'Edit' link) and 'Tags'. The 'Versions' section is expanded, showing a table of model versions. The table has columns: Version, Registered at, Created by, Stage, and Description. There are 7 versions listed, from Version 8 to Version 13. Each version has a checkbox, a green checkmark icon, and a 'None' button in the Stage column.

☐	Version	Registered at	Created by	Stage	Description
☐	Version 13	2022-12-09 14:31:45		None	
☐	Version 12	2022-12-09 14:15:15		None	
☐	Version 11	2022-12-09 11:51:15		None	
☐	Version 10	2022-11-09 12:50:02		None	
☐	Version 9	2022-11-09 12:35:43		None	
☐	Version 8	2022-11-08 16:22:13		None	

Figure 95: Model versioning details for 'Survival Rate' prediction study

In Figure 94 and Figure 95 we have a view of all model versions that were tested to assure the correct creation of one model that covers the "Survival Rate" scenario.

In Figure 96 and Figure 97 the model's metrics are presented in detail.

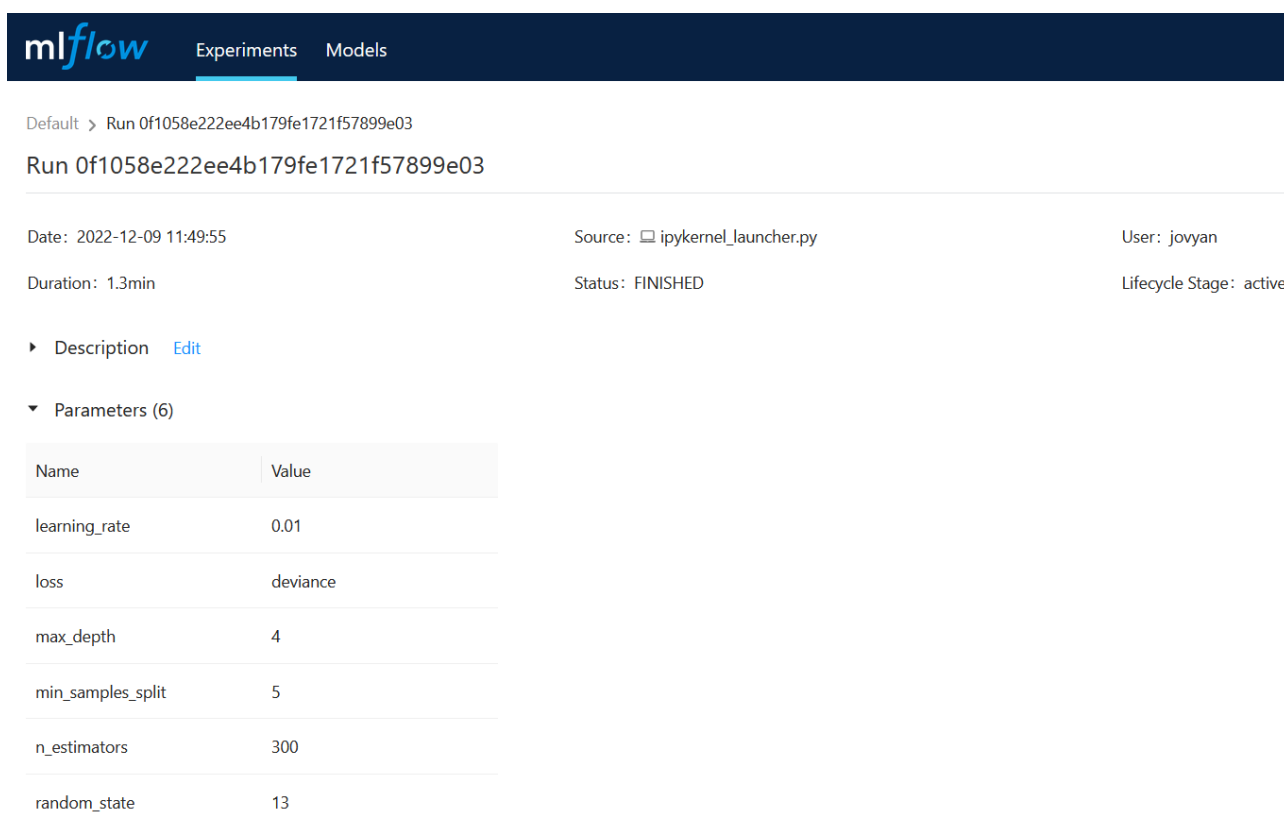


Figure 96: Model performance, KPI



Figure 97: Model's metrics

▼ Artifacts

▼ gbc-heart-failure

MLmodel

conda.yaml

python_model.pkl

requirements.txt

▶ model_explanations_shap

gbc-heart-failure-confusion-matrix.png

Full Path:s3://mlartefacts/0/0f1058e222ee4b179fe1721f57899e03/artifacts/gbc-heart-failure/MLmodel

Size: 300B

artifact_path: gbc-heart-failure

flavors:

python_function:

cloudpickle_version: 1.6.0

env: conda.yaml

loader_module: mlflow.pyfunc.model

python_model: python_model.pkl

python_version: 3.8.8

run_id: 0f1058e222ee4b179fe1721f57899e03

utc_time_created: '2022-12-09 09:51:12.528436'

▼ Artifacts

▼ gbc-heart-failure

MLmodel

conda.yaml

python_model.pkl

requirements.txt

▶ model_explanations_shap

gbc-heart-failure-confusion-matrix.png

Full Path:s3://mlartefacts/0/0f1058e222ee4b179fe1721f57899e03/artifacts/gbc-heart-failure/conda.yaml

Size: 213B

channels:

- conda-forge

dependencies:

- python=3.8.8

- pip

- pip:

- mlflow

- backports-abc==0.5

- cloudpickle==1.6.0

- psutil==5.8.0

- scikit-learn==1.1.2

- typing-extensions==3.7.4.3

name: mlflow-env

▼ Artifacts

▼ gbc-heart-failure

MLmodel

conda.yaml

python_model.pkl

requirements.txt

▶ model_explanations_shap

gbc-heart-failure-confusion-matrix.png

Full Path:s3://mlartefacts/0/0f1058e222ee4b179fe1721f57899e03/artifacts/gbc-heart-failure/requirements.txt

Size: 105B

mlflow

backports-abc==0.5

cloudpickle==1.6.0

psutil==5.8.0

scikit-learn==1.1.2

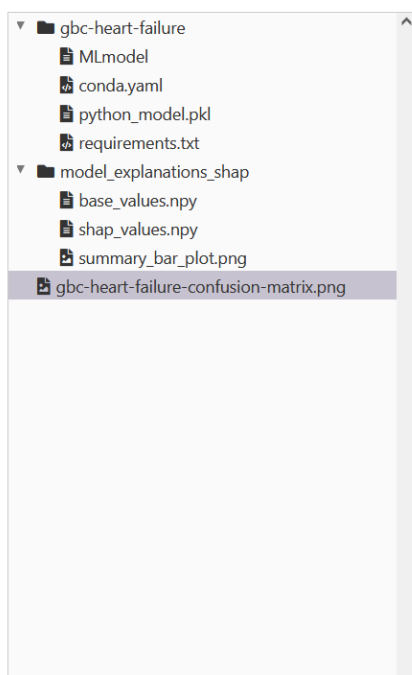
typing-extensions==3.7.4.3

Figure 98: Model artifacts

Here we can see the details of one model artifacts (Figure 98). Along with the “pkl” file which is actually the serialized model, there are several other types of information like the requirements, the environment in which the model was trained, Python version etc.

Besides this type of information we store attached to the model additional “meta-information” of the model. This is currently represented by the confusion matrix depicted in Figure 99 and the feature importance in model creation obtained using SHAP. The results from SHAP algorithm are presented in Figure 100.

▼ Artifacts



Full Path:s3://mlartefacts/0/0f1058e222ee4b179fe1721f57899e03/artifacts/gbc-heart-failure-confusion-matrix.png
Size: 6.91KB

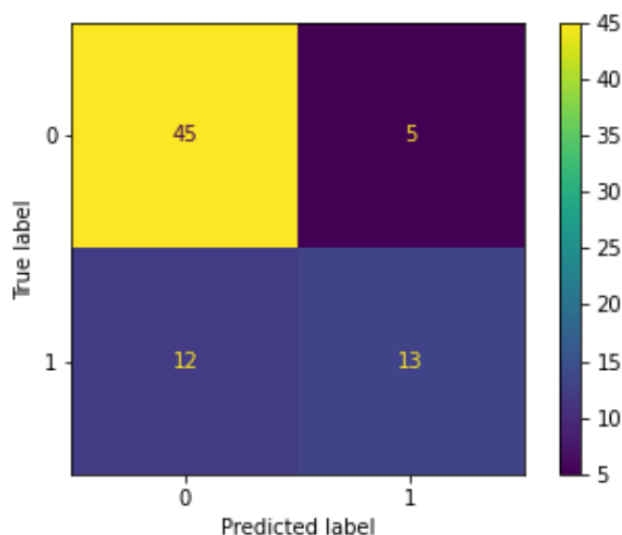
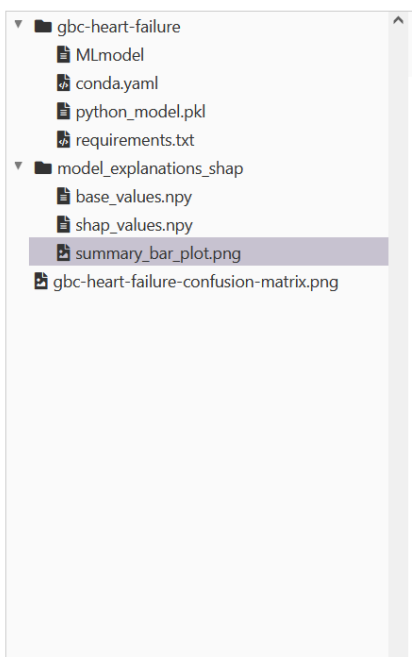


Figure 99: Confusion Matrix

▼ Artifacts



Full Path:s3://mlartefacts/0/0f1058e222ee4b179fe1721f57899e03/artifacts/model_explanations_shap/summary_bar_...
Size: 14.17KB

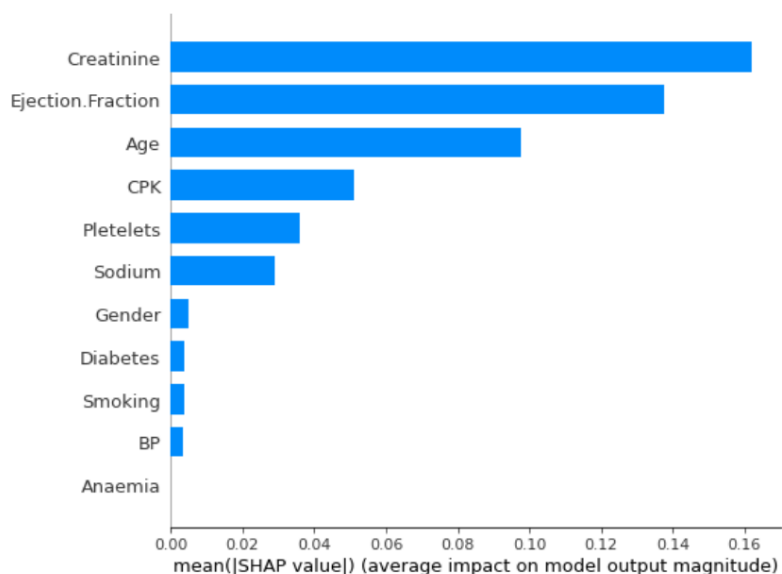
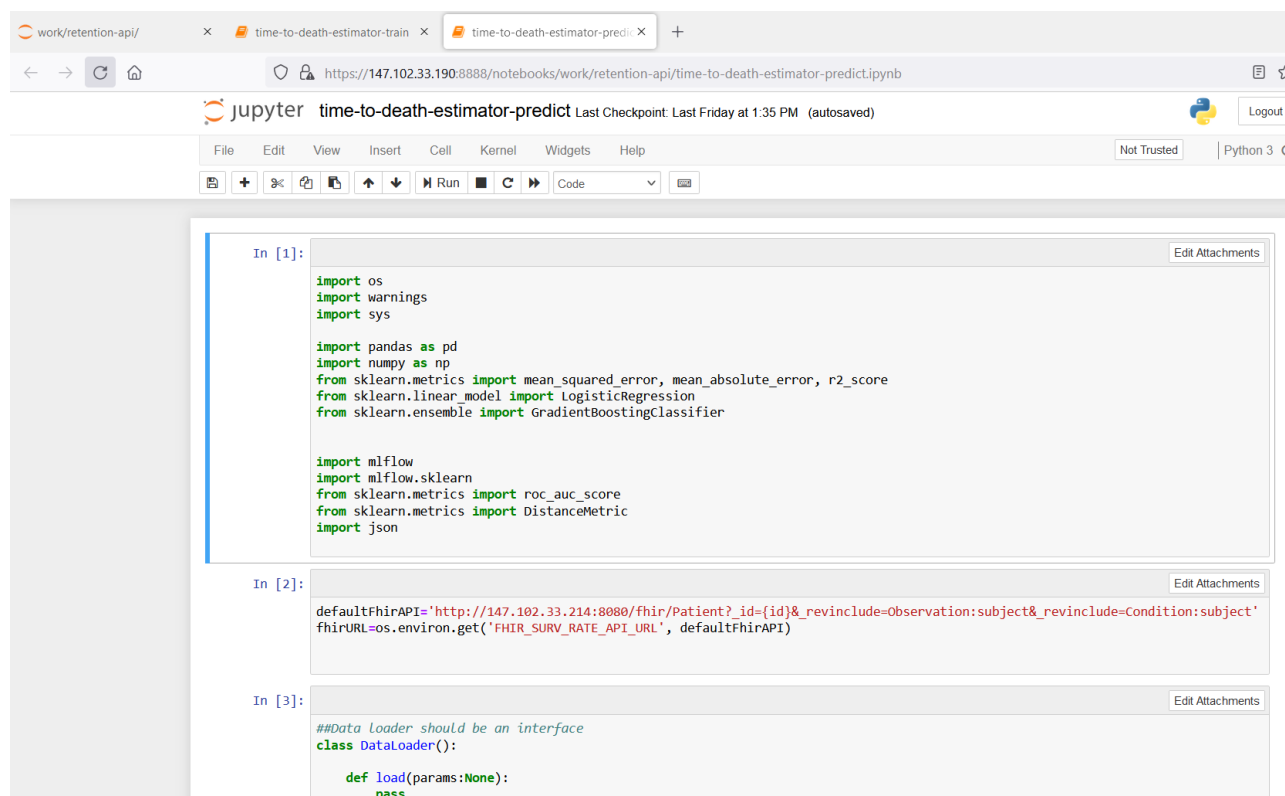


Figure 100: SHAP results

5.6 Using Jupyter Notebook

From data analysis perspective, the role of the data scientist is to read the specific data focusing on each developed scenario to create models that will serve later to clinicians.

In Section 4.3.2, we demonstrated the process of obtaining ML-based predictions. In Figure 101, we depict what the data scientist sees on GIC server, namely Python code in the Jupyter Notebook environment. This IDE facilitates up both the development as well as the deployment to CSB side.



```
In [1]:
import os
import warnings
import sys

import pandas as pd
import numpy as np
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import GradientBoostingClassifier

import mlflow
import mlflow.sklearn
from sklearn.metrics import roc_auc_score
from sklearn.metrics import DistanceMetric
import json

In [2]:
defaultFhirAPI='http://147.102.33.214:8080/fhir/Patient?id={id}&revinclude=Observation:subject&revinclude=Condition:subject'
fhirURL=os.environ.get('FHIR_SURV_RATE_API_URL', defaultFhirAPI)

In [3]:
##Data Loader should be an interface
class DataLoader():
    def load(params=None):
        pass
```

Figure 101: “Time to death” scenario from a data scientist perspective



6 Conclusions and next steps

The objective of the D6.1 was to demonstrate the work done within WP6 and the Tasks T6.1-T6.4 in specific. The first versions of the Mobile Application, the Local Home Gateway, the Clinical Site Backend and the Global Insights Clouds were presented and the main functionalities and workflows were explained through screenshots and complementary remarks. The required configuration and the involved APIs were also presented, realizing a working version of the interfacing components of the RETENTION platform. Deliverables D7.1 and D7.2 will present the integrated RETENTION platform in its entirety, including not only the interfacing components presented here, but also the components that support their operation.

This first version of the interfacing components described herein is fully functional and can serve the needs of the end users, focusing on the one hand on offering the patients easy ways of feeding the RETENTION platform from their home and wearable devices and on the other on clinicians' needs by providing a tool that allows them to monitor their patients and, also, perform various predictions and analyses that will potentially provide a better insight to the evolution of the patients' heart disease status.

Further enhancements, informed by user feedback and the Consortium's experiences from the RETENTION pilot, will result in the next version to be described in D6.3 (M36).



7 Appendix

7.1 Installation instructions for the Local Home Gateway

Below the installation instructions for the Local Home Gateway are summarised;

1. Download the clone image for the Raspberry Pi from GIC.
2. Flash the image to the micro SD card of the Raspberry Pi. A laptop with a micro SD card reader port or an external SD card reader to USB is required.
3. The proposed software for flashing the image are balenaEtcher for Linux, Win32 Disk Imager for Windows and ApplePi-Baker for macOS.
4. Insert the SD card into the Raspberry Pi.
5. Connect the Raspberry Pi 4 to the power, to the internet through an ethernet cable and to a monitor through a micro-HDMI to HDMI cable. In case of a Raspberry Pi 3 the cable required for monitor connection is HDMI to HDMI.
6. Insert a battery into the Xiaomi temperature and humidity sensor. If more Xiaomi temperature and humidity sensors are in range, make sure to remove their batteries. Bring the sensor close to the Raspberry Pi.
7. Double click the tool.py script on the Desktop and then click run.
8. Choose the option "Scan for sensor mac address" and then click the button "Scan". Wait for 15 seconds. Two mac addresses should appear, one for the sensor and one for the Raspberry Pi (packageId). Then click the button "Main Menu".
9. Choose the option "Parameters Configuration", and fill in all the following fields:
 - a. **UserId** is the pseudo-Id for the current patient.
 - b. **PackageId** is the mac address of the Raspberry Pi, acquired from the previous step.
 - c. **Mac Address** is the mac address of the sensor, acquired from the previous step.
 - d. **Country** requires the code of the country in the standard ISO 3166.
 - e. **Zip Code**
10. Choose the option "Choose Organization Id" and make a choice between the available pilot sites (virtual pilot of pilots, OCSC, NKUA, SERMAS - HURC, SERMAS - HUPH, UNIBO, UKESSEN).
11. Choose the option "Testing Process" and click the button "Fetch". Wait for 15 seconds. If every field is filled in with values the installation of the system is successful. If a field contains the value "Failed", then installation is not successful. If the "Failed" value refers to any of the sensor data (first three fields) then the problem is about the sensor and the procedure from step 6 and on, should be followed again. If the "Failed" value refers to any of the weather data (last three fields) then the problem is about the open weather map service and the procedure from step 9 and on, should be followed again.
12. Choose the option "Remote software installation" to install the software for remote access to the Raspberry Pi. After the installation is complete, create a custom password from the software's settings.
13. Choose the option "Disable screen". This option will disable the Raspberry Pi screen and reboot. After that the system will be ready. Make sure to successfully execute steps 11 and 12 before choosing this one.



7.2 API data

Temperature

```
{
  "resourceType": "Observation",
  "id": "Patientbodytemperature",
  "text": {
    "status": "generated",
    "div": "<div xmlns='http://www.w3.org/1999/xhtml'><p><b>Generated Narrative</b></p><div style='display: inline-block; background-color: #d9e0e7; padding: 6px; margin: 4px; border: 1px solid #8da1b4; border-radius: 5px; line-height: 60%'><p style='margin-bottom: 0px'>Resource 'Patientbodytemperature' </p></div><p><b>status</b>: final</p><p><b>code</b>: Patient body temperature <span style='background: LightGoldenRodYellow; margin: 4px; border: 1px solid khaki'> (<a href='https://browser.ihtsdotools.org/'>SNOMED CT</a>#386725007)</span></p><p><b>subject</b>: <a href='Patient/1'>Patient/1</a></p><p><b>effective</b>: 2021-05-21 09:29:23+0000</p><p><b>value</b>: 120</p></div>"
  },
  "status": "final",
  "code": {
    "coding": [
      {
        "system": "http://snomed.info/sct",
        "code": "386725007",
        "display": "Patient body temperature"
      }
    ]
  },
  "subject": {
    "reference": "PXXXX"
  },
  "effectiveDateTime": "2021-05-21T09:29:23.356+00:00",
  "valueQuantity": {
    "value": 120
  }
}
```

Pressure

```
{
  "resourceType": "Observation",
  "id": "Bloodpressuremeasurement",
  "text": {
    "status": "generated",
    "div": "<div xmlns='http://www.w3.org/1999/xhtml'><p><b>Generated Narrative</b></p><div style='display: inline-block; background-color: #d9e0e7; padding: 6px; margin: 4px; border: 1px solid #8da1b4; border-radius: 5px; line-height: 60%'><p style='margin-bottom: 0px'>Resource 'Bloodpressuremeasurement' </p></div><p><b>status</b>: final</p><p><b>code</b>: Blood pressure measurement <span style='background: LightGoldenRodYellow; margin: 4px; border: 1px solid khaki'> (<a href='https://browser.ihtsdotools.org/'>SNOMED CT</a>#75367002)</span></p><p><b>subject</b>: <a
```



```
href=\"Patient/1\">Patient/1</a></p><p><b>effective</b>: 2021-05-21
09:29:23+0000</p><p><b>value</b>: 120</p></div>\"
},
\"status\" : \"final\",
\"code\" : {
  \"coding\" : [
    {
      \"system\" : \"http://snomed.info/sct\",
      \"code\" : \"75367002\",
      \"display\" : \"Blood pressure measurement\"
    }
  ]
},
\"subject\" : {
  \"reference\" : \"PXXXX\"
},
\"effectiveDateTime\" : \"2021-05-21T09:29:23.356+00:00\",
\"valueQuantity\" : {
  \"value\" : 120
}
}
```

Oxygen saturation

```
{
  \"resourceType\" : \"Observation\",
  \"id\" : \"Theoxygensaturationpercentageofthepatient\",
  \"text\" : {
    \"status\" : \"generated\",
    \"div\" : \"<div xmlns=\\\"http://www.w3.org/1999/xhtml\\\"><p><b>Generated Narrative</b></p><div
style=\\\"display: inline-block; background-color: #d9e0e7; padding: 6px; margin: 4px; border: 1px solid
#8da1b4; border-radius: 5px; line-height: 60%\\\"><p style=\\\"margin-bottom: 0px\\\">Resource
\\\"Theoxygensaturationpercentageofthepatient\\\" </p></div><p><b>status</b>: final</p><p><b>code</b>:
The oxygen saturation percentage of the patient <span style=\\\"background: LightGoldenRodYellow;
margin: 4px; border: 1px solid khaki\\\"> (<a href=\\\"https://browser.ihtsdotools.org/\\\">SNOMED
CT</a>#442440005)</span></p><p><b>subject</b>: <a
href=\\\"Patient/1\\\">Patient/1</a></p><p><b>effective</b>: 2021-05-21
09:29:23+0000</p><p><b>value</b>: 120</p></div>\"
  },
  \"status\" : \"final\",
  \"code\" : {
    \"coding\" : [
      {
        \"system\" : \"http://snomed.info/sct\",
        \"code\" : \"442440005\",
        \"display\" : \"The oxygen saturation percentage of the patient\"
      }
    ]
  },
  \"subject\" : {
```



```
"reference" : "PXXXX"
},
"effectiveDateTime" : "2021-05-21T09:29:23.356+00:00",
"valueQuantity" : {
  "value" : 120
}
}
```

Weight

```
{
  "resourceType" : "Observation",
  "id" : "Bodyweight",
  "text" : {
    "status" : "generated",
    "div" : "<div xmlns=\"http://www.w3.org/1999/xhtml\"><p><b>Generated Narrative</b></p><div style=\"display: inline-block; background-color: #d9e0e7; padding: 6px; margin: 4px; border: 1px solid #8da1b4; border-radius: 5px; line-height: 60%\"><p style=\"margin-bottom: 0px\">Resource \"Bodyweight\" </p></div><p><b>status</b>: final</p><p><b>code</b>: Body weight <span style=\"background: LightGoldenRodYellow; margin: 4px; border: 1px solid khaki\"> (<a href=\"https://browser.ihtsdotools.org/\">SNOMED CT</a>#726527001)</span></p><p><b>subject</b>: <a href=\"Patient/1\">Patient/1</a></p><p><b>effective</b>: 2021-05-21 09:29:23+0000</p><p><b>value</b>: 120</p></div>"
  },
  "status" : "final",
  "code" : {
    "coding" : [
      {
        "system" : "http://snomed.info/sct",
        "code" : "726527001",
        "display" : "Body weight"
      }
    ]
  },
  "subject" : {
    "reference" : "Patient/1"
  },
  "effectiveDateTime" : "2021-05-21T09:29:23.356+00:00",
  "valueQuantity" : {
    "value" : 120
  }
}
```

Heart rate

```
{
  "resourceType" : "Observation",
  "id" : "HeartRateMeasurement",
  "text" : {
    "status" : "generated",
```



```
"div" : "<div xmlns=\"http://www.w3.org/1999/xhtml\"><p><b>Generated Narrative</b></p><div style=\"display: inline-block; background-color: #d9e0e7; padding: 6px; margin: 4px; border: 1px solid #8da1b4; border-radius: 5px; line-height: 60%\"><p style=\"margin-bottom: 0px\">Resource \"Heart rate measurement\" </p></div><p><b>status</b>: final</p><p><b>code</b>: Heart rate measurement <span style=\"background: LightGoldenRodYellow; margin: 4px; border: 1px solid khaki\">(<a href=\"https://browser.ihtsdotools.org/\">SNOMED CT</a>#364075005)</span></p><p><b>subject</b>: <a href=\"Patient/1\">Patient/1</a></p><p><b>effective</b>: 2021-05-21 09:29:23+0000</p><p><b>value</b>: 120</p></div>"
},
"status" : "final",
"code" : {
  "coding" : [
    {
      "system" : "http://snomed.info/sct",
      "code" : "364075005",
      "display" : "Heart rate measurement"
    }
  ]
},
"subject" : {
  "reference" : "PXXXX"
},
"effectiveDateTime" : "2021-05-21T09:29:23.356+00:00",
"valueQuantity" : {
  "value" : 120
}
}
```

Sleep

```
{
  "resourceType" : "Observation",
  "id" : "Sleepduration",
  "text" : {
    "status" : "generated",
    "div" : "<div xmlns=\"http://www.w3.org/1999/xhtml\"><p><b>Generated Narrative</b></p><div style=\"display: inline-block; background-color: #d9e0e7; padding: 6px; margin: 4px; border: 1px solid #8da1b4; border-radius: 5px; line-height: 60%\"><p style=\"margin-bottom: 0px\">Resource \"Sleepduration\" </p></div><p><b>status</b>: final</p><p><b>code</b>: Sleep duration <span style=\"background: LightGoldenRodYellow; margin: 4px; border: 1px solid khaki\">(<a href=\"https://browser.ihtsdotools.org/\">SNOMED CT</a>#762636008)</span></p><p><b>subject</b>: <a href=\"Patient/1\">Patient/1</a></p><p><b>effective</b>: 2021-05-21 09:29:23+0000</p><p><b>value</b>: 120</p></div>"
  },
  "status" : "final",
  "code" : {
    "coding" : [
      {
        "system" : "http://snomed.info/sct",
```



```
"code" : "762636008",
"display" : "Sleep duration"
}
],
},
"subject" : {
"reference" : "PXXX"
},
"effectiveDateTime" : "2021-05-21T09:29:23.356+00:00",
"valueQuantity" : {
"value" : 120
}
}
```

Distance

```
{
  "resourceType" : "Observation",
  "id" : "Distancewalkedperday",
  "text" : {
    "status" : "generated",
    "div" : "<div xmlns='http://www.w3.org/1999/xhtml'><p><b>Generated Narrative</b></p><div style='display: inline-block; background-color: #d9e0e7; padding: 6px; margin: 4px; border: 1px solid #8da1b4; border-radius: 5px; line-height: 60%'><p style='margin-bottom: 0px'>Resource 'Distancewalkedperday' </p></div><p><b>status</b>: final</p><p><b>code</b>: Distance walked per day <span style='background: LightGoldenRodYellow; margin: 4px; border: 1px solid khaki'> (<a href='https://browser.ihtsdotools.org/'>SNOMED CT</a>#246132006)</span></p><p><b>subject</b>: <a href='Patient/1'>Patient/1</a></p><p><b>effective</b>: 2021-05-21 09:29:23+0000</p><p><b>value</b>: 120</p></div>"
  },
  "status" : "final",
  "code" : {
    "coding" : [
      {
        "system" : "http://snomed.info/sct",
        "code" : "246132006",
        "display" : "Distance walked per day"
      }
    ]
  },
  "subject" : {
    "reference" : "Patient/1"
  },
  "effectiveDateTime" : "2021-05-21T09:29:23.356+00:00",
  "valueQuantity" : {
    "value" : 120
  }
}
```