SMARTBEAR

CALL H2020-SC1-FA-DTS-2018-2020

Trusted digital solutions and Cybersecurity in Health and Care

TOPIC DT-TDS-01-2019

Smart and healthy living at home

SMART BEAR

"Smart Big Data Platform to Offer Evidence-based Personalised Support for Healthy and Independent Living at Home"

D4.4 (D16) - Report on SMART BEAR Cloud Enabling Components v2

Due date of deliverable: 2022-12-31
Actual submission date: 2023-02-21

Grant agreement number: 857172

Start date of project: 01/09/2019

Lead contractor: CNR

Duration: 60 months

Project funded by the European Commission within the EU Framework Programme for Research and Innovation HORIZON 2020	
Dissemination Level	
PU = Public, fully open, e.g. web	✓
CO = Confidential, restricted under conditions set out in Model Grant Agreement	
CI = Classified, information as referred to in Commission Decision 2001/844/EC.	
Int = Internal Working Document	

D4.4 (D16) - Report on SMART BEAR Cloud Enabling Components v2

Editor

Anca Bucur (Philips)

Contributors

Mario Sicuranza (CNR)
Daniel Foulkes (Philips)
Erno Ledder (Philips)
Anca Bucur (Philips)
Valerio Bellandi (UMIL)
Paolo Ceravolo (UMIL)
Ernesto Damiani (UMIL)
Jonatan Maggesi (UMIL)
Samira Maghool (UMIL)
Gabriel Marques Tavares (UMIL)
Mattia Occhipinti (UMIL)
Nello Scarabottolo (UMIL)
Christos Kloukinas (CITY)
Qiqi Su (CITY)
Eleftheria Vellidou (NKUA)

Reviewers

Ioannis Kouris (ICCS)

The work described in this document has been conducted within the project SMART BEAR, started in September 2019. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 857172

©Copyright by the SMART BEAR Consortium.

Executive Summary

This document reports on the second version of the components of the SMART BEAR backend Cloud platform. The deliverable documents the backend components and the analytics, together with planned work of WP4 focused on the integration of the platform, the further development of the key components, and on the implementation and evaluation of the analytics approaches adopted by the project.

The main components of the SMART BEAR cloud platform include the database and its underlying information model, the clinical repository interfaces, the Big Data Engine, the synthetic data generation component, and the analytics. The analytics and personalization components leverage the SMART BEAR FHIR-based Information Model. This second version reports on the work carried out since the release of D4.2.

Contents

1 Introduction 8
1.1 Overview 8
1.2 Goals of the deliverable 8
2 SMART BEAR Cloud Enabling Components 9
2.1 Data Model Specification compliant with FHIR 9
2.1.1 Introduction 9
2.1.1.1 Overview on FHIR and available tools 9
2.1.1.2 Methodology used in the SMART BEAR project 10
2.1.2 Data model mapping 11
2.1.2.1 FHIR artefacts produced 14
2.2 BDA Engine 18
2.2.1 Function and tools 18
2.2.1.1 Data storing and management 19
2.2.1.2 Data analysis tasks 19
2.2.1.3 Data flow management and workflow orchestration 20
2.2.1.4 Data visualization 20
2.2.1.5 Platform management 21
2.2.1.6 Analytics Orchestration 21
2.3 Analytics 22
2.3.1 Data quality evaluation 22
2.3.2 Missing data imputation 22
2.3.2.1 Data Imputation workflow 23
2.3.2.2 Automation of data imputation workflow in order to find the best imputation algorithm 27
2.3.3 Sample size approximation 27
2.3.3.1 Sample size evaluation workflow 27
2.3.3.2 Using AUTOML for training step in Sample size evaluation workflow 28
2.3.4 Predictive models 28
2.3.4.1 Feature selection 28
2.3.4.2 Time Series Features Extraction 29
2.3.4.3 Continuous Learning 29
2.3.4.4 Use case scenario: Mild Depression 29
2.3.4.5 Use case scenario: Cardiovascular diseases 33
2.4 Analytics for Hearing Loss 36
2.4.1 Data 36
2.4.2 AI Model and Approach 37
2.4.2.1 Prediction Model 38
2.4.2.2 Explanation Model 40
2.5 Federated Learning Framework 41
2.6 AI models and interventions in the CVD scenario 42
2.6.1 Models 42

2.6.1.1 LSTM 42
2.6.1.2 Neural Tree model 43
2.6.2 Applications of these models to support specific use cases in SMART BEAR 43
2.6.2.1 Heart rate prediction during exercises 43
2.6.2.2 Sleep duration prediction 44
2.6.2.3 Deterioration prediction and Regimen adherence 44
2.6.3 Data 44
2.6.3.1 Current challenges 44
2.6.3.2 Possible solutions 45
2.6.3.3 Chosen solutions 46
2.7 Dashboard 46
2.7.1 Registration and login 46
2.7.2 Patient creation 48
2.7.3 Notebooks and analytics 49
2.7.4 Assessments 50
2.7.5 Device allocation 52
2.7.6 Tech support 53
3 Concluding Remarks 55
4 References 56

List of Figures

Figure 1 Differential Table for Hearing Aid Device 15
Figure 2 Differential Table for HA Observation 16
Figure 3 Differential Table 17
Figure 4. Workflows of data imputation demonstrate the steps to be followed for the Heuristic approach while the lower diagram refers to the experimental approach implemented in SMART BEAR. 23
Figure 5.Evaluation of imputation algorithms using fake missing values 25
Figure 6: Automated imputation workflow 26
Figure 7: The Automated Sample size workflow 28
Figure 8: Feature importance in the Mild Depression prediction model 33
Figure 9: Features importance in the CVD prediction using the PoP dataset 35
Figure 10:The calculated metrics for the cross-validate data 35
Figure 11. The architecture of the proposed model – attn-ED 38
Figure 12. SHAP Visualisation in the case of Personalised Explanation 41
Figure 13. SHAP Visualisation in the case of Global Explanation 41
Figure 14 LSTM Cell Architecture 42
Figure 15 Neural Tree Model 43
Figure 16 Walking distance in the dataset 45
Figure 17 Aggregated heart rate values 45
Figure 18. The registration form 47
Figure 19. Snapshot of signing page to access the Registration and Logging page 47
Figure 20. End user management page which enables the editing 48
Figure 21. The form for creating a patient 48
Figure 22. List of added Notebooks designed and launched by data scientists 49
Figure 23. A closer look at the Blood pressure prediction notebook. 49
Figure 24. The creation of analytics 50
Figure 25. Medical History of a patient. This information are stored by clinicians during the assessments. 50
Figure 26. List of questionnaires and scores for the patients under study. The edit and delete buttons are also active for modification by relevant clinician. 51
Figure 27. A view of the MoCA questionnaire used in the first assessment 51
Figure 28. Information on the devices assigned to a patient 52
Figure 29. Device assignment page 53
Figure 30. Technical support page to be managed by technicians. 53
Figure 31. A patient's overview page for the technicians 54
Figure 32. A screenshot showing the log files 54

List of Tables

Table 1. The variables for Mild Depression 30
Table 2: XGBOOST Model metrics on the train and cross-validation dataset. 31
Table 3: Confusion Matrix for train data 31
Table 4: Confusion Matrix for cross-validate data 32
Table 5: list of most important features in the predictive model of mild depression using augmented PoP data set followed by imputation workflow 32
Table 6. List of independent/dependant variable proposed for the Cardio Vascular Disease prediction 33
Table 7. Variables Names and Descriptions in the EvoSynth Dataset 36
Table 8. Hyper-parameters Settings 39
Table 9. Comparison of the Error Estimators Between attn-ED and Vanilla LSTM in the case of Personalised Prediction 40
Table 10. Comparison of the Error Estimators Between attn-ED and Vanilla LSTM in the case of Global Prediction 40

1 Introduction

1.1 Overview

Deliverable 4.4 Report on SMART BEAR Cloud Enabling Components v2 reports on the second version of the components that compose the SMART BEAR backend Cloud platform. This deliverable documents the infrastructure and analytics development activities part of WP4, which focuses on delivering all the components needed for the SMART BEAR Cloud infrastructure, including the data repository, the analytics models, the decision support system, the big data approaches, the developed interventions and their evaluation, and the visualisation.

The main components of the SMART BEAR cloud platform include the database and its underlying information model, the clinical repository interfaces, the Big Data Engine, the synthetic data generation component, and the analytics.

1.2 Goals of the deliverable

This deliverable provides the technical specification of the SMART BEAR cloud enabling components, describing their design, interactions, interfaces, and functionality. A core topic of this deliverable is the solution to data representation and management, including a full specification of the FHIR-compliant information model for clinical data, the description of the full data model (including clinical and non-clinical data) and of the database implementation.

We present approaches selected for the implementation of AI and analytics to support the development of effective interventions that will be deployed and evaluated in the SMART BEAR pilots. Each analytics approach is presented in the context of a specific clinical scenario, describing the targeted interventions, the process of developing and validating the model, and the used data.

2 SMART BEAR Cloud Enabling Components

2.1 Data Model Specification compliant with FHIR

2.1.1 Introduction

2.1.1.1 Overview on FHIR and available tools

FHIR is a standard created by HL7 International for exchanging digital health information by exploiting the best characteristics of HL7 Version 2 and Version 3, although it is not compatible with them. The standard is spreading rapidly in the international context and the most important international organizations that provide solutions to specific problems in healthcare, like Integrating the Healthcare Enterprise (IHE) and Personal Connected Health Alliance (PCHA), are updating their technical specifications to include FHIR.

FHIR defines the key health entities as resources, which are a collection of information models specifying data elements, constraints and relationships for the "business objects". The last stable FHIR specification published is Release 4, version 4.01, which is based on the FHIR Composition Framework that includes 145 different resource types covering many clinical and administrative concepts of the healthcare sector. Each resource type, like *Patient*, *Observation*, and *Device*, permits to represent a number of properties related to a specific concept. In order to facilitate a homogeneous representation of the data into these resources, the FHIR specification provides a concrete formalization of them in three different data formats: eXtensible Markup Language (XML), JavaScript Notation Object (JSON), and Turtle (TTL). In FHIR, many elements are used to memorize terminologies represented by coded values, like the element *code* of the *Observation* resource. Some of these codes are directly defined by the FHIR specification, while other codes may be defined elsewhere, like: external terminology systems.

Another important component of the FHIR specification is represented by the RESTful Application Programming Interfaces (APIs), which are a collection of well-defined interfaces for making different applications able to interoperate. Their use is useful to use shared communication protocols via Hypertext Transfer Protocol (HTTP), even if not mandatory.

The FHIR specification is a "platform specification", as it describes a large number of base resources, frameworks and APIs. However, in order to implement a FHIR-based solution for a specific subdomain of the healthcare able to take into account different regulations, requirements, etc., the FHIR specification requires further adaptations to the particular context. These adaptations include a specification of the particular standard resource elements that will be used, possible additional elements, the APIs and terminologies to use, etc. Several FHIR artefacts may be used for adapting the specification to a specific context: i) Implementation Guide (IG), which is a set of rules well documented that specify how FHIR resources have to be used to solve a particular problem; ii) Package, which is a group of related adaptations published within an IG; iii) Conformance Resource, which is a specific FHIR resource in a package that represents the adaptations in a computable way.; iv) Profile, which is a set of constraints and extensions on a single resource.

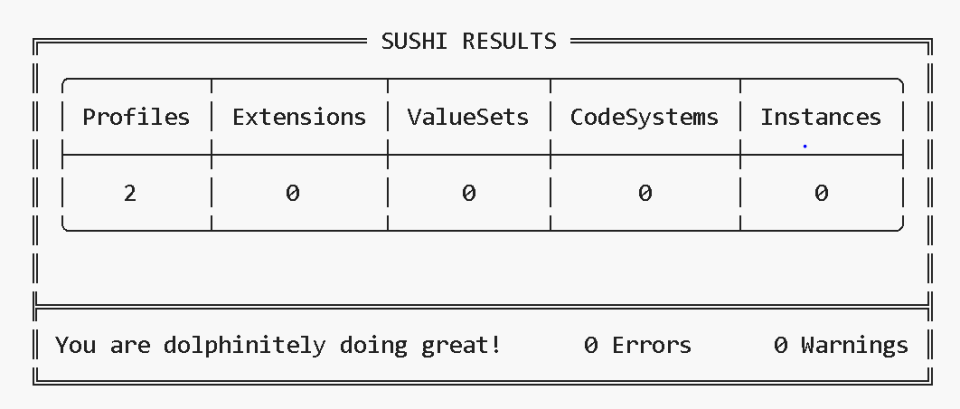
A number of frameworks and tools exist for facilitating the implementation and adaptation of solutions in compliance with the FHIR standard.

Simplifier.net is a web platform for publishing, collaborating and developing the FHIR specification, offering capabilities for resource management and team collaboration. The platform provides a multitude of available FHIR resources organized in projects, consisting of profiles, extensions, value sets, dictionaries, mappings, examples and more. Forge can be used for creating FHIR profiles through a Windows desktop application, in an integrated way with Simplifier.net.

ART-DÉCOR is an open source tools suite that supports the creation and maintenance of standard HL7 templates, value sets, scenarios, and datasets. It therefore represents both a methodology for the creation of standard artefacts and a set of tools to simplify the operations of managing artefacts. The tool also includes the

management of the templates and value sets through cloud-based federated Building Block Repositories and therefore allows to simplify the entire phase of use and reuse of published templates and value sets. ART-DECOR supports collaboration between members of the same team or between different teams through governance operations.

FHIR Shorthand (FSH) is a language specified by HL7 for defining the contents of the FHIR IGs. It permits to express resource profiles, value sets, examples, etc. using any text editor, enabling thus distributed, team-based development using source code control tools such as GitHub. It is based on a formal grammar defined in the ATLR parser generator. SUSHI Unshortens ShortHand Inputs (SUSHI) is a FSH compiler, which convers scripts written according to FSH language to FHIR artefacts. As shown in the nex figure, SUSHI permits to compile and produce artefacts concerning profiles, extensions, value sets, code systems and instances.



IG Publisher is a tool provided by the FHIR team for translating the IG content in: i) generated resources that can be included into the published guide (XML, JSON, TTL formats); ii) HTML files; iii) zip files used by implementers. The tool is used after the compilation of the FSH scripts by the SUSHI compiler.

ART-DÉCOR and FSH are free, whereas Simplifier.net has both free and paid profiles and Forge is free only for educational purposes.

2.1.1.2 Methodology used in the SMART BEAR project

The nature of the data treated in the project, in accordance with the rules of the FHIR standard, led to the necessity of using a specific IG. For this reason, an analysis of the IGs published on the FHIR registries was carried out. Among these, particular attention was dedicated to the Personal Health Device (PHD) and International Patient Summary (IPS) IGs.

The PHD IG adapts FHIR resources to convey measurements and supporting data from PHDs to different kind of systems, like platforms for electronic medical records, clinical decision support, etc. The interest for this IG was captured considering that it is based on the Continua Design Guidelines and upon the ISO/IEEE 11073 PHD Domain Information Model (DIM). Anyway, considering that in SMART BEAR many health data are not acquired by PHDs, but by collecting patients' answers to specific questionnaires, this IG was not considered adequate for the SMART BEAR project.

The IPS IG defines the rules for the production of a document containing the essential healthcare information about a subject of care, designed for supporting unplanned, cross border care, although it is not limited to it. Although this IG provides an important contribution to identify a minimal, specialty-agnostic, condition-independent, clinically relevant dataset for a patient, it was not considered relevant for the SMART BEAR project.

For these reasons, in compliance with the FHIR standard and in line with the choices adopted in many European projects, the approach taken for the definition of the information model for the project is to define a dedicated SMART BEAR IG by profiling a set of identified FHIR resources and individuating the terminologies from international standard code systems as well as internal value sets. The tools chosen for modelling the FHIR

information model are FHIR Shorthand, SUSHI and IG publisher, considering that they are the official tools provided by HL7.

Starting from Information model we have defined and published a dedicated Smart Bear FHIR Implementation Guide, using the HL7 tools.

In particular, the tools used were:

- FHIR Shorthand (FSH)
  it is a domain-specific language for defining the content of FHIR implementation guides.It allows resource profiling, value sets, examples, using any text editor. Use formal grammar.
- SUSHI (Unshortens ShortHand Inputs)
  it is a reference implementation and de facto standard for a FSH compiler that translates FSH into FHIR artifacts such as profiles, extensions, and value sets. It therefore allows you to convert scripts written according to the FSH language into FHIR artifacts.
- IG Publisher
  It is a tool provided by the FHIR team to translate IG content into:
  - generated resources that can be included in the published guide (XML, JSON, TTL formats);
  - HTML files;
  - zip files used by implementers.

The tool was used after the FSH scripts were compiled by the SUSHI compiler. And then get the different web pages of the IG. These tools are the official tools provided by HL7 international.

2.1.2 Data model mapping

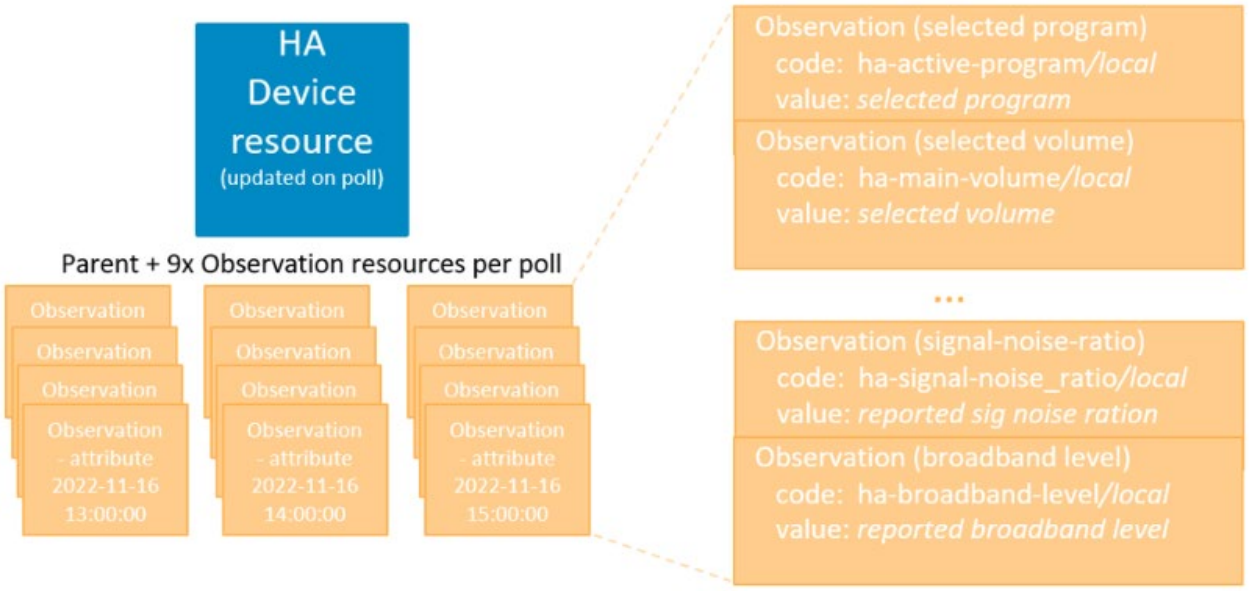
This section gives the specification of the data collected, and the associated constraints and extensions to the base FHIR model. In a subsequent iteration, this specification will be formalized in a FHIR Implementation Guide. Where applicable, relevant profiles and definitions are mentioned.

The rest of the section details the SMART BEAR patient information, the mapping to and constraints on FHIR resources, relevant profiles and notes. Here we extend the model described in the previous version of this deliverable (which was D4.2 Report on SMART BEAR Cloud Enabling Components v1), and focus on the hearing loss use case, which was not covered in D4.2.

Hearing aid monitoring takes place by means of an App that polls the hearing aids for data via Bluetooth. Polling takes place on regular basis (nominally 15 min), and the data derived from the hearing aids is formed into a set of FHIR Observation resources each of which contains a single aspect of the hearing aid data. All of these Observation resources are linked to a single “parent” Observation resource that maintains the collection of information. All Observation resources reference the Device resource that defines a specific hearing aid device. Properties of the Device resource are also updated with the latest polled information to provide a current view of the hearing aid status.

FHIR Resource Design for Hearing Aid Monitoring

- Device resource for each HA updated (stores point in time data)
- Multiple light weight Observation resources created for each poll
- Properties stored in individual Observation resources (eg. A parent observation resource plus one for each of 9 attributes)



Hearing Aid Properties:

Code	Definition/Notes	Related Value (Observation)	Related Property (Device)
property-set	Code for the "parent" Observation to reference a set of component Observations.		N/A
mute-status	Code for Observation containing mute status of the hearing aid. In Device this is represented by statusReason=paused.	valueBoolean with options: true=muted | false=unmuted	N/A
main-volume	Main volume setting of the hearing aid.	valueCodableConcept.coding[0].code with options: ["-4","-3","-2","-1","0","1","2","3","4"]	valueCode[0].coding[0].code with options: ["-4","-3","-2","-1","0","1","2","3","4"]
main-program	Main program setting of the hearing aid.	valueCodableConcept.coding[0].code with options: ["0","1","2","3"]	valueCode[0].coding[0].code with options: ["0","1","2","3"]
time-since-boot	Time since last boot in seconds.	valueInteger	valueQuantity[0].value

total-usage-time	Total time device has been used in seconds.	valueInteger	valueQuantity[0].value
signal-noise-ratio	Signal to Noise Ratio Estimator.	Array of decimal values in: component[n].valueQuantity.value	Array of decimal values in: valueQuantity[n].value
noise-level	Noise Level Estimator.	Array of decimal values in: component[n].valueQuantity.value	Array of decimal values in: valueQuantity[n].value
broadband-level	Broadband Level Estimator	Single decimal value in: valueQuantity/value	Single decimal value in: valueQuantity[0]/value

Device resource data elements:

Element	Contents
status	Normally "active". Use "inactive" if the device is no longer operational or a part of the project
statusReason	"**online**" if device can communicate, "**paused**" for muted state, or "**offline**" if device is not accessible
manufacturer	Name of device manufacturer
type	Left HA "**6012004:272741003=7771000**" or Right HA "**6012004:272741003=24028007**"
property[]	Set of properties as outlined in the Related Property (Device) column of the Hearing Aid Properties table
patient	Reference to related Patient resource

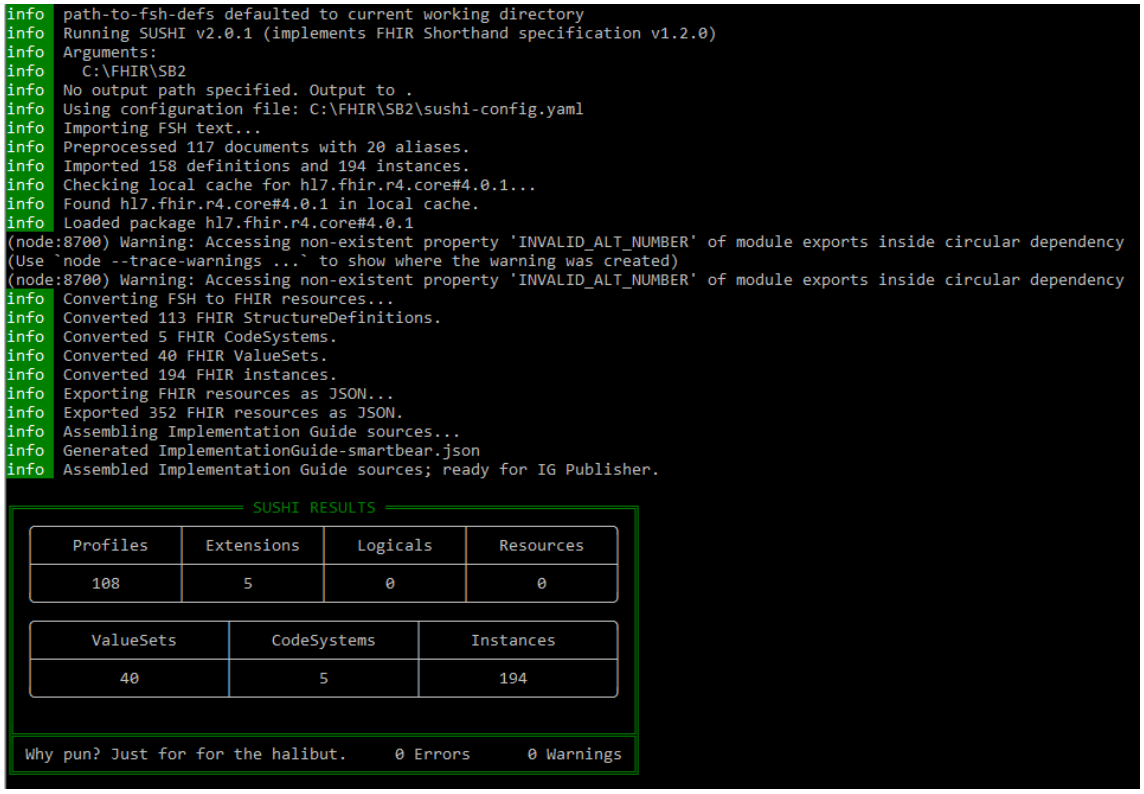
The Observation resource data elements are described in the table below:

Element	Contents
status	Normally "active". Use "inactive" if the device is no longer operational or a part of the project
code	Item from the Code column of the Hearing Aid Properties table indicating the value of this Observation
subject	Reference to the related Patient resource
effectiveDateTime	Date time of observation
bodySite	Left Ear "**89644007**" or Right Ear "**5577004**"
device	Reference to the relevant Device resource
hasMember[]	Only required when code == 'property-set': a set of references to the hearing aid Observations taken at the same time
value[]	The item in the Related Value (Observation) column of the Hearing Aid Properties table relevant to the code value of this resource.

2.1.2.1 FHIR artefacts produced

The specifications described in the previous sections have been translated in a formal FHIR IG using FSH and IG Publisher. The corresponding web pages have also been created.

Currently, the published IG (Implementation Guide) consists of 108 Resource Profiles (of type Observation, Condition, Questionnaires, Bundle, Patient, DeviceUseStatement, FamilyMemberHistory, MedicationStatement, ResearchSubject), 5 Extensions, 40 Value Sets and 198 examples. The FHIR IG is available at https://smart-bear.eu/fhir/ .



This section shows some illustrations of the web specifications produced for the Hearing Aids FHIR Artefacts. The main profiles produced are listed below.

Profile HADevice

It is a Device for Hearing Aid Device, left or right, the parent HL7 resource is Device.

The figure shows the Differential Table.

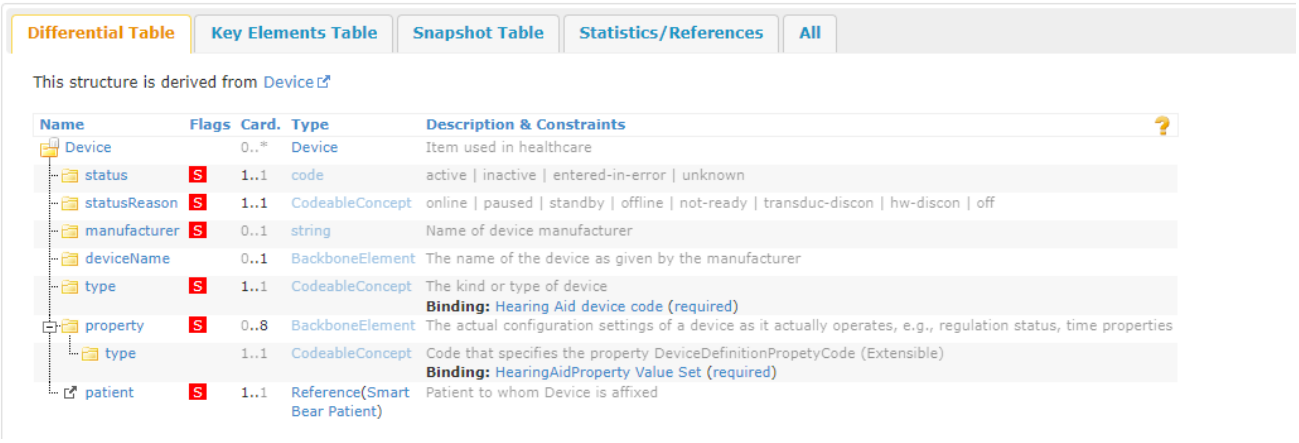


Figure 1 Differential Table for Hearing Aid Device

The type of Device is an CodeableConcept, where the possible code are:

- 6012004:272741003=7771000Left hearing aid
- 6012004:272741003=24028007Right hearing aid

The type of property is an CodeableConcept, wher the possible values are showed in the table below.

Code	System	Display	Definition
property-set	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Hearing Aid Property Set	Code for the "parent" Observation to reference a set of component Observations.
mute-status	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Mute Status	Code for Observation containing mute status of the hearing aid. In Device this is represented by statusReason=paused.
main-volume	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Volume	Main volume setting of the hearing aid
main-program	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Selected Program	Main program setting of the hearing aid
time-since-boot	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Time Since Boot	Time since last boot in seconds.
total-usage-time	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Total Usage Time	Total time device has been used in seconds.
signal-noise-ratio	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Signal to Noise Ratio	Signal to Noise Ratio Estimator.
noise-level	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Noise Level	Noise Level Estimator.

broadband-level	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Broadband Level	Broadband Level Estimator
rel-location	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Relative Location	Location relative to a specified home zone

ParentResourceHAObservation

It is an Observation for Parent Resource Hearing Aids Observation

The figure shows the Differential Table.

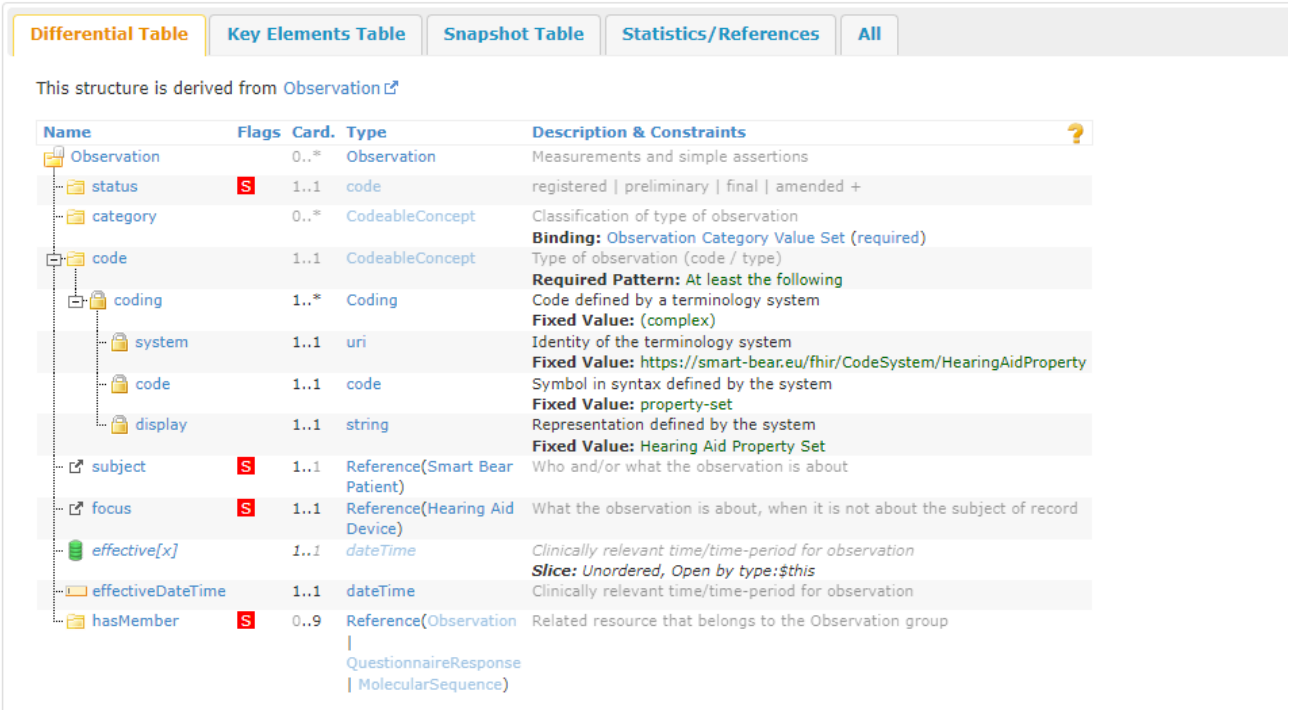


Figure 2 Differential Table for HA Observation

MemberResourceHAObservation

It is an Observation for Member Resource Hearing Aids Observation

The figure shows the Differential Table.

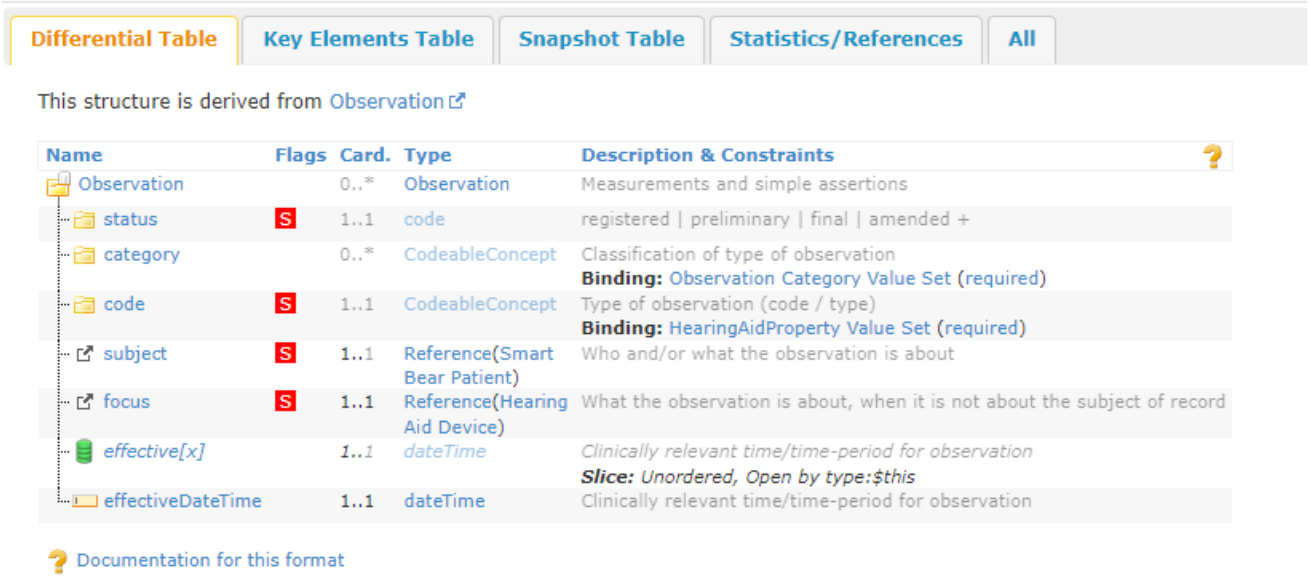


Figure 3 Differential Table

The code of this Observation is an CodeableConcept, wher the possible values are showed in the table below.

Code	System	Display	Definition
property-set	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Hearing Aid Property Set	Code for the "parent" Observation to reference a set of component Observations.
mute-status	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Mute Status	Code for Observation containing mute status of the hearing aid. In Device this is represented by statusReason=paused.
main-volume	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Volume	Main volume setting of the hearing aid
main-program	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Selected Program	Main program setting of the hearing aid
time-since-boot	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Time Since Boot	Time since last boot in seconds.
total-usage-time	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Total Usage Time	Total time device has been used in seconds.
signal-noise-ratio	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Signal to Noise Ratio	Signal to Noise Ratio Estimator.

noise-level	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Noise Level	Noise Level Estimator.
broadband-level	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Broadband Level	Broadband Level Estimator
rel-location	https://smart-bear.eu/fhir/CodeSystem/HearingAidProperty	Relative Location	Location relative to a specified home zone

2.2 BDA Engine

The BDA engine mainly addresses the functionalities required for processing DAWs (Data Analysis Workflows) and providing/storing execution results.

2.2.1 Function and tools

The BDA uses a series of appropriately configured Open-Source components and a custom developed one that is responsible for piloting the execution of the various analyzes available in the catalog and for saving the results and then showing them in the dashboard.

The Components used by the platform are the following ones:

- *Apache Hadoop*: of the Hadoop ecosystem we use almost exclusively **HDFS**[1], or the distributed file system. On **HDFS** we save the data that is extracted, for example, from the FHIR repository and brought in a tabular format through an appropriate transformation
- *Apache Hive Metastore*: it takes care of containing all the information regarding the databases, the tables and the relationships between them[2]. This is especially useful for handling data that is transformed and saved on the distributed file system. The metastore allows interoperability between the various components that need to access the data and the ability to then be able to launch SQL-like queries using Trino and Spark
- *Apache Spark:* it is the component that allows us to launch data engineering, data science and machine learning tasks on large datasets[3]. Thanks to him we can access the data through the metastore considering the HDFS data as if they were tables in a SQL database, simplifying life for those who have to develop the analytics and guaranteeing the necessary scalability for the platform
- *Trino:* it is a distributed SQL query engine designed to query large data sets distributed over one or more heterogeneous data sources[4]. It is designed to handle data warehousing and analytics: data analysis, aggregating large amounts of data and producing reports (OLAP). It can be used to do some ETL transformations without the overhead of Spark and with the proper configuration can run SQL query directly on tables stored in HDFS

[1] https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html

[2] https://hive.apache.org/

[3] https://spark.apache.org/

[4] https://trino.io/

- *Apache Airflow:* used to develop, schedule and monitor batch-oriented workflows[5]. Airflow's extensible Python framework enables us to build workflows using different technology, combined with docker it can run custom images with all the tools needed to run analytics
- *BDA API:* internally developed, it provides a REST-type interface to the dashboard and other platform components in order to be able to interact with workflows and save/retrieve the results of previous executions
- *Delta Lake:* open-source storage framework that enables building a Lakehouse architecture[6]. It sits on top of Hadoop. Delta Lake provides ACID Transactions, scalable metadata handling, and unifies streaming and batch data processing on top of existing data lakes like HDFS
- *Apache Zeppelin*: it is a Web-based tool that enables users to create interactive data analytics, prototype some of the analytics that should be translated in a proper workflow, share preliminary results in a collaborative environment for the data scientist[7].

2.2.1.1 Data storing and management

Data that is exported from various storage sources, such as the FHIR repository, is transformed through a series of steps and saved using the Delta Lake format. This type of format allows you to build a Lakehouse architecture that is an open architecture that combines the best elements of data lakes and data warehouses, some of the key features of this kind of architectures are the following ones:

- ACID Transactions
- Schema enforcement
- Business intelligence support
- Openness
- Support for diverse workloads
- Support for structured and unstructured data

Through Delta Lake we can support all these requirements, but with additional benefits. One of the benefits is the possibility to Time Travel to a precise version of a table. This allows us to be able to relaunch an analysis on a specific version of the data in order to be able to reanalyze and possibly make a comparison between the results of different snapshots. It also provides a mechanism to replicate a result, audit the data and the results obtained. The other benefit is the possibility to update or delete data. This is a key feature since the implementation of Delta Lake is based on Apache Parquet, a columnar storage format that takes advantages of efficiently compressing the data, but has the problem that doesn't support Delete and Upsert operations since it is a format that can only Append data to a file. The Delta Lake implementation therefore uses a metadata file to support all this and through a series of commands allows us to easily switch from a Delta-type format to normal data in Parquet format.

2.2.1.2 Data analysis tasks

Data analysis tasks are mainly performed through the use of Spark and Python libraries that can operate on data in Delta Lake format. Depending on the complexity of the workflows, these can be composed of several steps, in some cases it is easier to query and prepare data through Spark and then use other more specific libraries or

[5] https://airflow.apache.org/

[6] https://delta.io/

[7] https://zeppelin.apache.org/

tools to perform the required analytics. With Apache Airflow we can create DAGs (directed acyclic graph) composed of different tasks and use HDFS as a distributed file system to save the data needed for the various steps, furthermore this allows us to be flexible since, if necessary, it is possible to use also ready-made Docker images containing all the tools necessary to create models and perform analytics using the most appropriate tools. The main requirement therefore always remains to be able to read data from a distributed file system such as HDFS.

2.2.1.3 Data flow management and workflow orchestration

The most complex job within data flow management is the one that queries the FHIR repository to export the data from the repository and insert it into the tables in Delta Lake. The workflow, that runs each day, is therefore composed of different tasks which can be roughly divided as follows:

- Export from FHIR
- Flattening of the data
- Insert in Delta Table

In order to export the data, the possibilities offered by the FHIR REST API are exploited, using the Bulk Data Export API. it is therefore possible to request to export data that has been inserted/modified within the repository since the last time the export process was successful. The data exported in *JSON* format are then saved on HDFS, the next step is to bring them in a flat type format, i.e., as if they were data belonging to a common SQL table. The FHIR structures are complex JSON and therefore contain both arrays and nested objects within them. In order to bring them to the desired shape, an ad hoc script was therefore created using Apache Spark which takes care of this and consists of the following steps:

1. *Flatten all the properties*: it means that each property and object nested in the JSON creates a column by concatenating the name of the properties with those that are children of the main property. For example, a *code_coding_code* is created which is the corresponding path *code.coding.code* which can be found within a common FHIR resource.
2. *Create a new row for each element of an array*: If the element contains arrays within the structure in this case a new row is created in the resulting table for each element of this array, an example of this behavior can be found within the Observation used to represent the measured Blood Pressure during the assessments.

All of this is orchestrated using Apache Airflow as the main engine. Using Airflow, it is possible to configure the workflows to be executed according to predefined intervals, either with some already predefined, or in very specific cases it is possible to execute them also using schedules defined through an interval defined through a Cron type expression. This task uses some of the main features of Airflow, such as the ability to launch workflows according to a predefined time interval and use the last date the task was successfully executed as the reference date for the export from FHIR repository. A benefit deriving from the fact of using Apache Airflow is that of being able to express the workflow structures as code and therefore the possibility of integrating them within the CI/CD cycle without necessarily having to force manual updates with the possibility therefore of losing track the version of the workflow that is deployed in the system

2.2.1.4 Data visualization

For the Data Visualization part, the BDA Engine mainly makes use of two different tools: *Apache Echarts* to display the results of the analytics in the Dashboard interface and *Apache Zeppelin* to allow data scientists to carry out data exploration and try to create examples of analytics which will then have to be deployed in production. Apache Echarts is an open-sourced JavaScript visualization tool that can run on Web Browser and mobile devices, it also provides an excellent library of basic charts and the possibility to extend or customize

according to the needs of our output[8]. There are many components available, among the most used there are certainly:

- *Datazoom* that is used for zooming a specific area, which enables user to investigate data in detail, or get an overview of the data, or get rid of outlier points
- *Timeline* which provides functions like switching and playing between multiple Echarts
- *Toolbox* that contains some functionalities like the export to PNG and zooming
- *Legend* that shows symbol, color and name of different series. The user can click legends to toggle displaying series in the chart

In order to provide an easy way to sketch analytics and share some preliminary results between data scientist the BDA Engine uses Apache Zeppelin. Through Zeppelin it is possible to launch Python, Spark, SQL Query or R scripts. This is made possible by the different interpreters that are provided by the system and that are launched using Docker images containing the required components and libraries already configured to execute the snippets. If necessary, the data scientist can also install other libraries in order to have all the tools required to run and plot the results obtained using different libraries. After creating a notebook, you can share the results obtained with other users of the platform, even in read-only mode, to obtain feedback and then move on to the analytics deployment phase in the system.

2.2.1.5 Platform management

Since all the tools used within the BDA Engine have been containerized, to manage the entire platform the tool we rely on is the deployment platform itself, i.e., Kubernetes (K8s). K8s is the system that we use to handle scaling, automatic system deployment and to manage all the containerized applications that we use in the Cloud[9]. All the components described above have their basic configurations saved as *ConfigMap* inside the Smart Bear repository dedicated to the BDA Engine while passwords and sensitive data are saved as *Secrets*. For some components it was decided to use Helm Charts to deploy as for the Spark cluster, while for others ad hoc deployments are used. Then the final configurations should be done using the WEB UI provided by the tool. These administration UIs are not available to all users, but only to system administrators who can also modify the deployment of the various tools and manage the resources needed to keep the infrastructure fast enough.

2.2.1.6 Analytics Orchestration

The analytics that can be performed with custom configuration by clinicians through the dashboard are exposed and managed through the BDA REST API. These APIs expose a catalogue of the available analytics with their various configuration parameters and their default values. Every time an analytic is launched these parameters are passed to Airflow which takes care of managing the execution. In the UI, the clinician can therefore create a new notebook and within this notebook launch different analyses (or perhaps even the same one, but with different parameters) and then analyse the results which are returned at the end of the execution. In case of problems, the system administrator can always check the various runs using the Airflow Web UI and check in the logs where the problem occurred, if it is due to a wrong configuration, lack of resources or something else. Obviously for the clinician or data scientist the management of the platform is totally transparent, in fact the configuration of the capacity of the scheduler and the number of workers available are tasks left to the system administrator who must find the right compromise between the available resources and the speed of execution requested by the end user.

[8] https://echarts.apache.org/en/index.html

[9] https://kubernetes.io/docs/home/

2.3 Analytics

This section describes the range of approaches to analytics and AI leveraged in the various clinical scenarios of SMART BEAR.

2.3.1 Data quality evaluation

The sparse multivariate temporal series collected by smart devices in smart-health projects requires specific methodologies to leverage feature sparsity by effectively studying the correlation between features, the achieved data quality levels, and the appropriate imputation techniques to be adopted. The proportion of missing data only provides limited information about the bias and efficiency gains that can be made from data imputation. There are five traits to be considered within data quality criteria: Accuracy, Completeness, Reliability, Relevance, and Timeliness.

In this list, *Accuracy* refers to the fact that received data are correct in every details. Regarding the *Completeness* trait, the missing values are taken into account. Data comprehensiveness is crucial in analysis validity. Contradiction with other data set is the *Reliability* trait to take into account. Whilst *Relevance* trait concerns how this data is really needed for our analysis. Then *Timeliness* refers to measuring how much the data are up-to-date and whether they are usable for real-time reporting. In the SMART BEAR infrastructure accuracy and timeliness are evaluated in terms of out-of-range data, using a pruning step to identify sensors sending data outside the expected domain or an expected timeframe. This record level assessment is not providing any information about the deficiency of the entire data set, for completing data quality evaluation, data imputation task will take place. The recent literature clarified that data imputation needs to be anticipated by correlation analysis to verify the pattern followed by missing values [18]. This implies that researchers should consider whether all the variables related to missingness can plausibly be included in the imputation model to limit bias and improve accuracy. An additional challenge of imputation techniques in longitudinal studies is related to the consistent integration of temporal series and scalar features in imputation models [19]. Similar considerations apply to continuous monitoring and prognostic models, Machine Learning algorithms require to get in input multidimensional vectors where all the sources of information are represented in terms of numerical data.

2.3.2 Missing data imputation

Due to different reasons such as technical problems or user ignorance in using the device properly, many data points may be missed during the study. The most general patterns identified in the literature are Missing Not At Random (MNAR), Missing At Random (MAR), and Missing Completely At Random (MCAR). MNAR, means there is a relationship between the propensity of a value to be missing and its values. For example, people with the lowest education are not answering to questionnaires including the questionnaire on their educational courses. MAR, means missing values are not in relation with other missing values but are in relation with observed values. For example, when men are more likely to report their weight than women. MCAR, means there is no relationship between missing values and any other values. Nothing makes some data more likely to be missing than others. For example, blood pressure records are missing randomly, due to non-adherence to the protocols of users but also due to blood pressure monitors batteries going out of charge. Recent studies found that, compared to restricting the analysis to individuals with complete data, imputation techniques improved the accuracy of predictions at any proportion of missing data. This implies that researchers should consider whether all the variables related to missingness can plausibly be included in the imputation model to limit bias and improve accuracy.

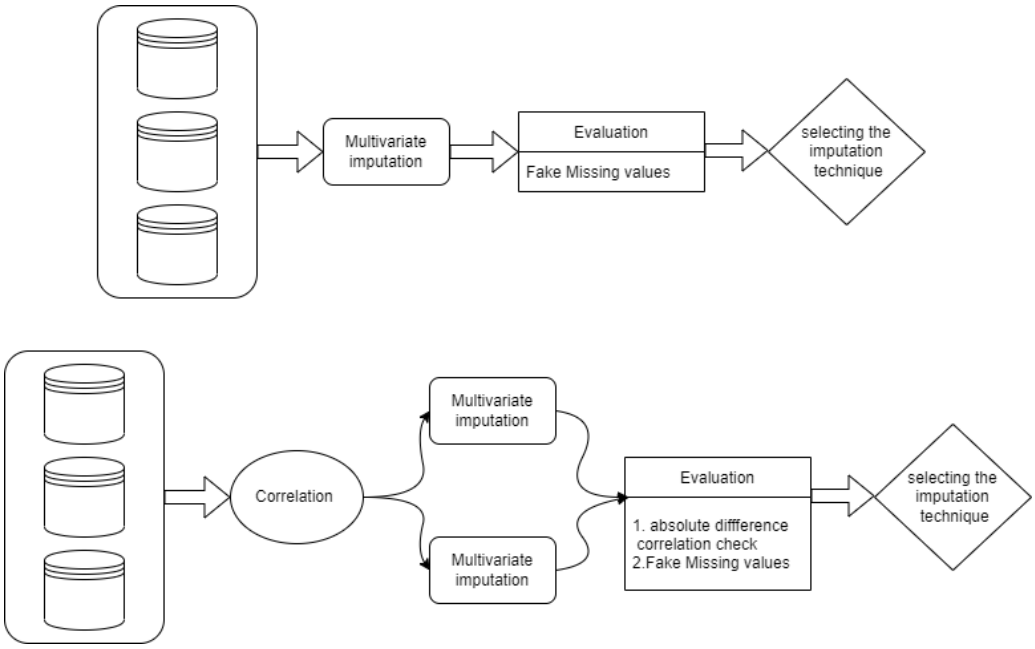


Figure 4. Workflows of data imputation demonstrate the steps to be followed for the Heuristic approach while the lower diagram refers to the experimental approach implemented in SMART BEAR.

2.3.2.1 Data Imputation workflow

Similarly, to other workflows in SMART BEAR project, the imputation is also needed to be implemented repetitively as new data are arriving at the repositories. Considering the fact that the data set contains, numerical, categorical and, time series values we need to follow the homogenization steps described earlier. Numerical values are normalized to one using the standard scale to minimum and maximum values. Categorical values are mapped to numerical while taking into account the null values as a number, for example, if we have yes/no/null values, we map them to 0/1/2. After the data imputation stage, the reverse mapping and then the one-hot encoding will be applied to the data set. One of the main concerns in the data imputation process is the loss of significance of data set caused by the imputed values. For this reason, we need to revise the data distribution after imputing data using quantitative values such as a correlation matrix. In this workflow, we aim to keep the correlation of the variables as much as possible similar to their correlation before the imputation process. In this workflow, considering the size of received data set while focusing on the correlation matrix, we propose two generic approaches in order to select the proper algorithm data imputation (see Figure 4):

- Heuristic: The literature has clarified it is not easy to classify the distribution of missing data, it requires having a deep understanding of the domain but most of the methods start from the assumption that distribution is MAR and a correlation with other observed data is possible, then the obtained accuracy is measured. In this approach, first, the correlation matrix of features in data set is calculated. If there are any correlated features, different “multivariate" algorithms will be performed on the data set imputing the missing values, otherwise, we continue with “univariate" algorithms. Univariate algorithms such as, mean, most-frequent value and moving average while the Multivariate algorithms such as regression, stochastic regression, XGboost, MICE and miss forest are considered for imputation purposes
- Experimental: A priori classification of the distribution of missing data is not only difficult but also impossible once there are multiple sources of missingness. Therefore, testing experimentally the different methods seems a necessity. In the experimental approach we propose: 1) Compute the correlation matrix in the data set, 2) Apply multiple methods of data imputation considering the

correlated variables, 3) Verify which one is producing the correlation matrix most similar to the original correlation matrix or in other words the absolute difference of two correlation matrices before and after imputation is converging to zero, 4) Using it applies again multiple methods, 5) Iterate until the variation of the correlation matrix before and after imputation is negligible.

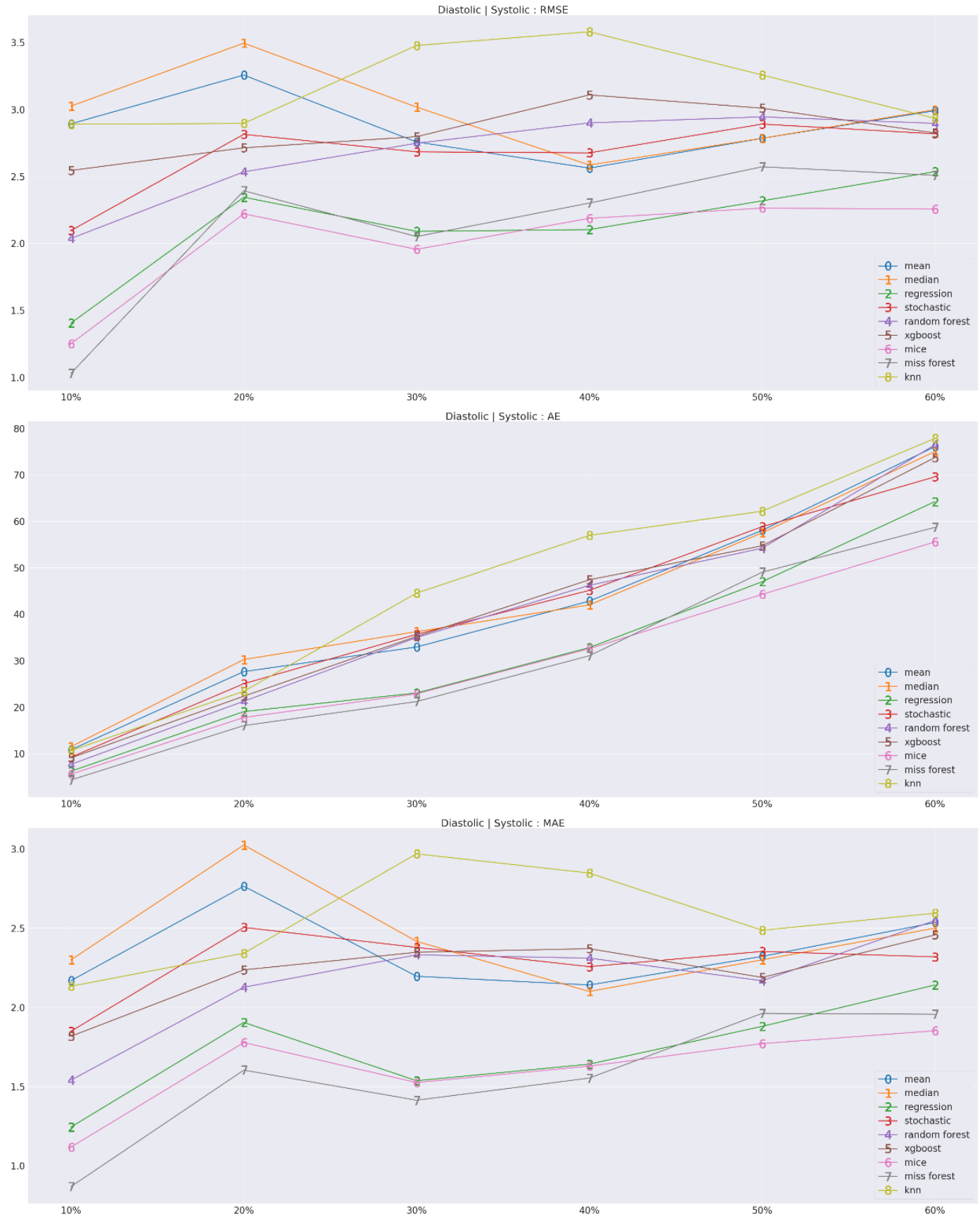


Figure 5.Evaluation of imputation algorithms using fake missing values

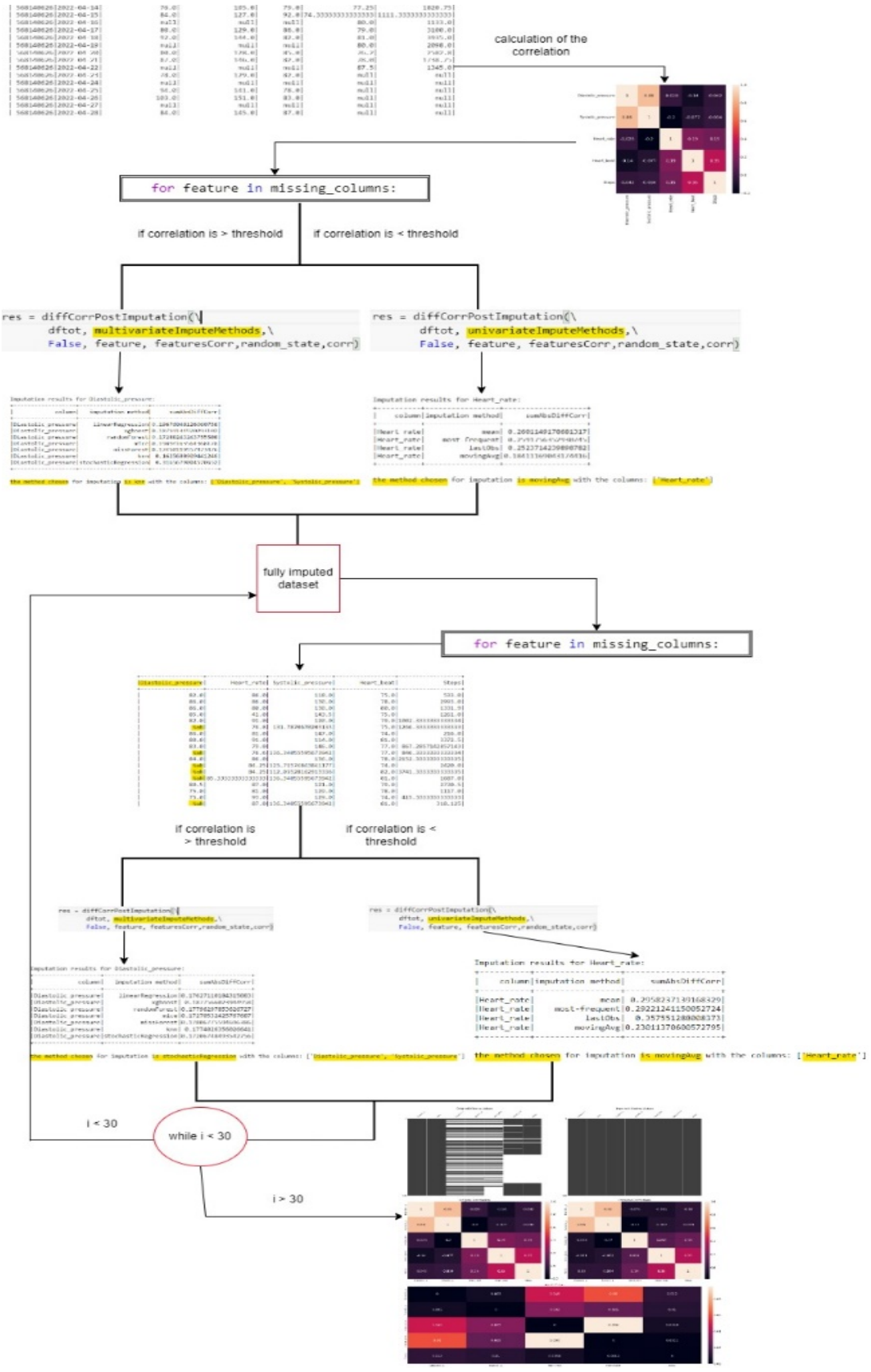


Figure 6: Automated imputation workflow

2.3.2.2 Automation of data imputation workflow in order to find the best imputation algorithm

We propose a library that automatically returns the imputed dataset with proper algorithms considering the missing values distribution. The library returns the completed dataset with least variation in data distribution that is validated by the fake missing values algorithm (see Figure 5. Initially, we calculate the correlation matrix and choose the uni/multivariate algorithm based on the correlated variables. Having the fully imputed dataset in hand, we try to make the imputed values more and more accurate by continuously evaluating one feature over other imputed features. This is also to exclude any unfavoured dependency between variables. In this step taking one by one those correlated features, as it was before, with its null values, while considering the rest feature completed in the previous step. The correlation matrix is again calculated and similarly to how we proceed initially in choosing uni/multivariate algorithms based on the threshold value for the absolute difference in correlation values before and after each time of imputation, we try to impute this specific column with at least 5 different algorithms. At this step, if the difference in correlation stays below the threshold, we keep the imputed values.

2.3.3 Sample size approximation

Predicting the volume of needed data points for validating ML model is one of the crucial aspects of initiating a study with limited resources. For this purpose, SMART BEAR project defines a three-stage procedure for the definition, implementation, and validation of prognostic models. First, at the homogenization process, multidimensional data such as temporal series are unified into a single-valued representation. Second, correlation analysis supports the identification of the patterns followed by missing data, to identify correlated features as their integration into machine learning models can support imputation and predictions. Third, the fraction of missing information is studied to predict the impact of missing value on the prognostic models tested by the project. The sample size must be evaluated with a temporal plan to define a credible timing for getting realistic volumes of data. We cannot plan using complex models if the data size is not appropriate. For example, experimental evidence from the literature [17] shows Random Forest is reliable with a sample size of 200 times the number of features in the feature space. This means once we know the sample size, we can also identify the dimension of the feature space to be used and the modelling tool to select.

2.3.3.1 Sample size evaluation workflow

Similarly, to the data imputation approaches, we take two approaches for sample size evaluation:

- Heuristic. Due to the fact that the ML algorithms need a high volume of data to train reasonably accurate models, and according to the existing literature, at least 200 times of the number of features are required as training data set for Random Forest classifier. Acquiring this size of the dataset in studies such as SMART BEAR containing many features while limited resources for recruiting many participants is not feasible. In this sense, we propose an experimental approach, for approximating the needed sample size to train ML models.
- Experimental. in this approach we consider following steps:

1) We start augmenting dataset by a defined proportion
2) We implement the imputation workflow dealing with the missing values
3) A Machine Learning model, such as a classifier, will be trained
4) As a solution for the model validation, cross-validation or external validation using the open data set will take place.
5) After validating the model, accuracy in the predicted values is discussed with the clinicians.
6) In order to further clarification and provide more insights, the feature relevance also will be discussed with clinicians.

7) In the case that model accuracy and relevant features of the model are approved by clinicians, we proceed with the next steps, otherwise we continue augmenting the initial data set.
8) After approval by clinicians, we extract the most important features and back to the inspiration by heuristic approach, we multiply the number of features by 200 and roughly calculate the needed number of data points for the accurate model.

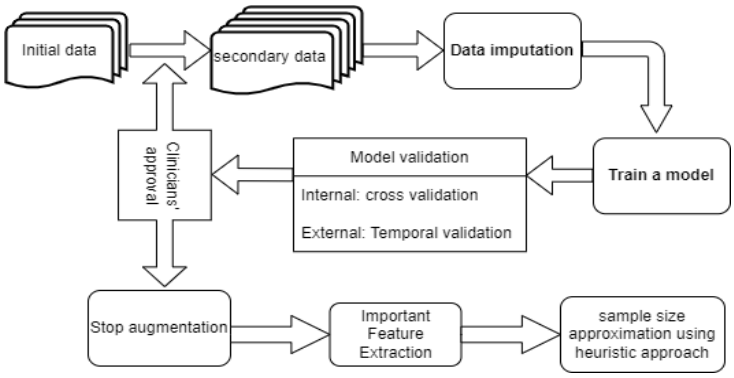


Figure 7: The Automated Sample size workflow

2.3.3.2 Using AUTOML for training step in Sample size evaluation workflow

AutoML[10], is the process of automating the iterative tasks of machine learning models implementations. It allows data scientists, analysts, and developers to build ML models with efficiency, and productivity while sustaining model quality. Traditional machine learning model development is costly in term of resources and time to produce and compare dozens of models while leveraging automated ML models the time needed for modelling significantly decrease. As a modification to our contribution in Sample size evaluation workflow, we leverage the AutoML in the training step. This way we make sure to tune the model efficiently (see Figure 7).

2.3.4 Predictive models

2.3.4.1 Feature selection

To build a prognostic algorithm model, one can select variables or features based on clinical knowledge and hypotheses, or using a hypothesis-free approach using all available data. In the SMART BEAR project, clinicians provide a list of relevant dependent/independent variables for studying each comorbidity. Most models for clinical studies rely on supervised learning. In supervised learning, the model is developed using gold standards defined by a clinical expert, for example, a chart review containing 200 patients with and, without Cardiovascular Diseases with the goal to identify the pattern that exists between patients with cardiovascular diseases vs those without. Unsupervised ML models can also be used for this purpose when there is a need to combine multiple conditions. These models are generally not as accurate as supervised models, but enable high-throughput phenotyping over a handful to thousands of models with improved accuracy over diagnosis codes alone. Moreover, unsupervised models have been used to build clinical models to predict disease courses, optimise diagnostics and target treatment. Unsupervised pattern recognition analyses identify subgroups of patient-patient similarity in a high dimensional or graph-based space.

[10] https://docs.h2o.ai/h2o/latest-stable/h2o-docs/automl.html

2.3.4.2 Time Series Features Extraction

Extraction of features of time series may take place for different purposes. In some cases, due to data size, analysing time series data set is highly resource demanding. Therefore, working on distributed machines or multiple cores leveraging the features of time series rather than the actual data set is very common. In other cases, such as training ML algorithms using time series data, while we encounter heterogeneous data types, extracting features as numerical values makes modeling more feasible to implement. As another usability of feature extraction, increasing the accuracy of models having more features as input to ML model. In Sections 3.3.4.4 and 3.3.4.5 we present a demonstrative example, where we use the dataset related to the CVD and try to train a classifier with and without the features of the time series measures. The Average accuracy with a subset of data equals to 0.3390. Adding the extracted features of timeseries to the initial data set we get the Average of accuracy equal to 0.7979. moreover, the important features in training this classifier, have slightly changed. While in the initial model, age and physical activity have high importance, after adding time series features, Heart beat mean and Diastolic pressure quantile 0.5 are also important in the predicted result.

2.3.4.3 Continuous Learning

Regarding the objectives of a project aiming the monitoring and collecting the observational data, the learning approach is not a one-shot task. This process may be improved while the project is running and new data are arriving to be exploited to improve data quality. Furthermore, the Model validation also will be applied in a repetitive fashion. During the observational study, data gathering, and analysis are taking place. Therefore, relating to the type of analysis, the newly arrived data set could be utilized for the model validation similar to external validation. Due to the fact that the new data will arrive with a time delay, temporal validation also could be possible considering the volume of arrived data. Regarding the ML models, the new dataset could play the test set role while the validation to find the appropriate parameters leverages the k-fold cross-validation algorithm. Therefore, the mentioned workflows, such as sample size approximation could take into account newly arrived data set as test set taking part into the continuous learning process while leveraging the AutoML in tuning the hyperparameters for adjusting the more accurate model.

2.3.4.4 Use case scenario: Mild Depression

Considering the fact that there are very few solutions on the market dedicated to early detection or/and prevention in the mental health field, focusing actively on the depression factors which could be considered as early signs or risk factors of clinical depression is crucial.

In SMART BEAR we study Mild Depression through 12-month continuous monitoring of the participants by collecting of heterogeneous data, such as demographics, mood status and sleep quality, cognitive status, exercise habits, social behaviour, adherence to medications, through various sensors, smartphone and face-to-face assessments and remote counselling with the medical practitioners involved in the project (see Table 1)

In SMART BEAR, considering different case studies suggested by clinicians, as we mentioned, a set of possibly relevant dependent and independent variables are defined in order to respond to the need of early diagnosis.

Table 1. The variables for Mild Depression

Dependent variables	Independent variables	Independent variables (from our interventions)
GDS PHQ-9	• Age: numerical • Gender: categorical • Civil Status: categorical • Educational Status: categorical • Household size: numerical • Financial Status: categorical • Smoking (yes/no): binary variable • Alcohol intake (yes/no): binary variable • Supplement usage: categorical • History of Cardiovascular Diseases (yes/no): binary variable • History of Mild Cognitive Impairment (yes/no): binary variable • Other medical history: categorical • Current medication use: categorical (types of meds) and numerical (number of meds) • Body Mass Index (BMI): numerical • Med Diet adherence (yes/no): binary variable • SQS: numerical	• Recommendations and visit doctor (yes/no): binary variable • Bodyweight: numerical (measurement and weekly average) • Physical activity: numerical (weekly average of active minutes) • Social interactions: numerical (weekly average of moving out of home)

Following the proposed workflows for the data imputation, we proceed with the workflow of Sample size evaluation using AutoML. At this workflow, after augmenting the existing data by 10%, we proceed with the imputation of all missing values using the automated algorithm we proposed earlier. Then we try to train models using AutoML to foster an efficient model by tuning hyper parameters with the augmented dataset. In order to internal validation the trained models, we consider at least 5-folds in cross-validation. It is worth noting that, in the prepossessing step of data, we consider the scaling using standard scaling and mapping the categorical data into numerical ones in order to validate the imputation process (after imputation data with be mapped back to the original form).

After imputation step and evaluating the imputed values using the Absolute difference in the correlation metric before and after imputation, and fake missing values we get a fully imputed dataset to proceed. In this example

we use the H2O[11] in favour of modelling automation by XGBoost[12] estimator. The statistics on the train set is as demonstrated in tables Table 3, Table 5 and Figure 8.

Table 2: XGBOOST Model metrics on the train and cross-validation dataset.

ModelMetricsBinomial: xgboost **** Reported on train data. ****	**ModelMetricsBinomial: xgboost** **** Reported on cross-validation data. ****
MSE: 0.0405 RMSE: 0.2014 LogLoss: 0.1512 Mean Per-Class Error: 0.0 AUC: 1.0 AUCPR: 1.0 Gini: 1.0	MSE: 0.17271 RMSE: 0.4155 LogLoss: 0.5203 Mean Per-Class Error: 0.222 AUC: 0.722 AUCPR: 0.306 Gini: 0.444

Table 3: Confusion Matrix for train data

Confusion Matrix (Act/Pred) for train data	**no**	**yes**	**Error**	**Rate**
no	3.0	6.0	0.0759	(6.0/72.0)
yes	.0	80.0	0.0	(0.0/80.0)
Total	73.0	86.0	0.0377	(0.0/85.0)

[11] https://h2o.ai/

[12] https://xgboost.readthedocs.io/en/stable/

Table 4: Confusion Matrix for cross-validate data

Confusion Matrix (Act/Pred) for train data	no	yes	Error	Rate
no	43.0	36.0	0.4557	(36.0/79.0)
yes	1.0	14.0	0.0667	(1.0/15.0)
Total	44.0	50.0	0.3936	(37.0/94.0)

Considering the best model in accuracy which is about 0.78, we extract the most important features in the acquired accuracy. The following table demonstrate a list of those features and the relevance values.

Table 5: list of most important features in the predictive model of mild depression using augmented PoP data set followed by imputation workflow

variable	relative_importance	scaled_importance	percentage
SQS	62.8780861	1.0	0.2487780
BMI	50.1625290	0.7977744	0.1984688
Household	37.6687241	0.5990756	0.1490368
History of Mild Cognitive Impairment	33.4897118	0.5326134	0.1325025
Educational Status	31.7515221	0.5049696	0.1256253
Smoking	20.6827221	0.3289337	0.0818315
Age	6.8156543	0.1083947	0.0269662
History of Cardiovascular Diseases	4.6966934	0.0746952	0.0185825
GDS	2.5024848	0.0397990	0.0099011
Gender	2.0996110	0.0333918	0.0083071

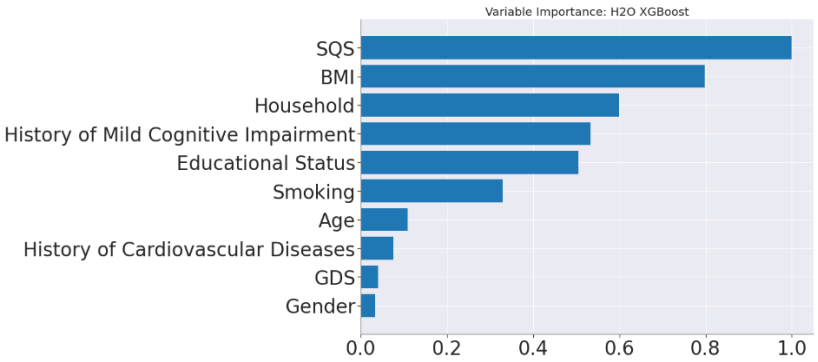


Figure 8: Feature importance in the Mild Depression prediction model

2.3.4.5 Use case scenario: Cardiovascular diseases

Following the proposed workflows for the data imputation and sample size evaluation for this dataset, we proceed with predictive model for the Cardio Vascular Diseases. Regarding a simplified model for the prediction of Cardio/non-cardio state of patients as labels to be returned by a model, we select the relevant variables proposed by clinicians (see Table 6).

Table 6. List of independent/dependant variable proposed for the Cardio Vascular Disease prediction

Independent variables	**Dependent variables**
• Weight • Ambient temperature • Height • Sleep estimation • BMI • LDL levels baseline • HDL levels baseline • triglyceride levels baseline • total cholesterol levels baseline • HBA1c levels baseline • PHQ-8 results • Edmonton Frailty Scale (EFS) • Ham Anxiety scale • Hip circumference • Waist circumference • Mediscore at recruitement • MNA at recruitement • Systolic Blood pressure at recruitement sitting/lying • Systolic Blood pressure at recruitement standing • Diastolic Blood pressure at recruitment sitting/lying • Diastolic Blood pressure at recruitment standing • Resting Heart Rate at recruitment sitting/lying • Resting Heart Rate at recruitement standing • HeartScore (0 months, 6 months, 12 months) • Smoking status • Alcohol intake	• number of non-scheduled visits due to hypertensive peaks (Urgency / Emergency Hypertension) during the intervention period (6 months, 12 months) • percentage of adherence in prescribed medical treatment • personalised aerobic physical activity per week. • Number of non-scheduled visits due to volume overload in subjects with heart failure during the intervention period (baseline for previous year, 6 months, 12 months) • Optimal BP control (personalised) expressed as time in therapeutic range (TTR) • Adoption of recommended dietary habits (mediscore change 6, 12 months) • Body mass index change • Weekly average systolic blood pressure level difference from personalised optimal • Weekly average diastolic blood pressure level difference from personalised optimal • Number of persistent extreme systolic BP measurements (6 months, 12 months) • Number of outside the optimal range systolic BP • Number of outside the optimal range diastolic BP • Number of persistent extreme diastolic BP measurements (6 months, 12 months) • Resting heart rate daily average within optimal range • Minimum resting heart rate • Maximum resting heart rate • Average daily heart rate (smartwatch) • Maximum daily heart rate (smartwatch) • Minimum daily heart rate (smartwatch) • Arrhythmia events (number per month of follow up) • HeartScore change

• Salt intake • Supplement usage – categorical • number of non-scheduled visits due to hypertensive peaks (Urgency / Emergency Hypertension) during the intervention period (baseline) • Number of non-scheduled visits due to volume overload in subjects with heart failure during the intervention period (baseline for previous year) • Number of non-scheduled admissions for any CVD reason 12 months prior to baseline • History of falls • Number of falls • Weekly average physical activity (minutes) at baseline • History of HTN • History of pacemaker • History of implantable defibrillator • History of permanent AF • History of paroxysmal AF • History of HFrEF • History of HFpEF • History of CAD • History of revascularization history • MNA - numerical • MOCA test • EuroQuality of Life • eGFR baseline • baseline ECG o PR interval (ms) o QRS interval (ms) o QTc (Bazet's) interval (ms) o Rhythm – options 1. Sinus rhythm 2. Paced rhythm 3. Atrial fibrillation 4. Atrial flutter o QRS morphology – options 1. Normal 2. LBBB 3. RBBB 4 LAH 5 LPH o LVH (yes/no)	• MNA change • Falls during intervention period (6, 12 months) • eGFR change (6, 12 months) • admissions for HF decompensation • admissions for unstable CAD • admissions for revascularization • admissions for AF • LDL level change (6, 12 months) • HDL level change • Total cholesterol level change • Triglyceride level change • HBA1c level change (6, 12 months) • Smoking status at 6, 12 months • Alcohol intake at 6, 12 months • Salt intake at 6, 12 months • 6, 12 months ECG o PR interval (ms) o QRS interval (ms) o QTc (Bazet's) interval (ms) o Rhythm – options 1. Sinus rhythm 2. Paced rhythm 3. Atrial fibrillation 4. Atrial flutter o QRS morphology – options 1. Normal 2. LBBB 3. RBBB 4 LAH 5 LPH o LVH (yes/no) o Change (6, 12 months) of above parameters • Time spent in light and moderate exercise (physical exercise tracker/heart rate bands – individualised – max Heart rate=220 – age in years. Moderate level = 64-76%) • Systolic and diastolic weekly averages per mean environmental temperature • Heart rate variability – time domain, smartwatch • Blood pressure variability (average real variability (ARV), variation independent of mean (VIM) , and residual standard deviation (RSD), coefficient of variance (CV)) per week, month and 6months • Blood pressure TTR per sleep quality • Blood pressure variability indices per Ham Anxiety Scale • Blood pressure TTR per HRV change • Physical activity per systolic blood pressure TTR • Physical activity per diastolic blood pressure TTR • Sleep quality/duration per systolic blood pressure TTR • Sleep quality/duration per diastolic blood pressure TTR • Sleep quality/duration per HRV change • Sleep quality/duration per BPV change • Blood pressure variability per depression scores • Blood pressure control (TTR) per depression scores • Mood assessment scores per diet change – additional variables duration, BP in range percentage, heart rate variability change, BMI change

From these variables some of them do not exist in the repositories so in this mock example we leverage those are already stored. Data curation process is implemented according to the mentioned workflows in the previous sections. In continue, by training the Random Forest on the 80% of the data set, we get the Average accuracy = 0.6212 with the important features as presented in Figure 9 and Figure 10.

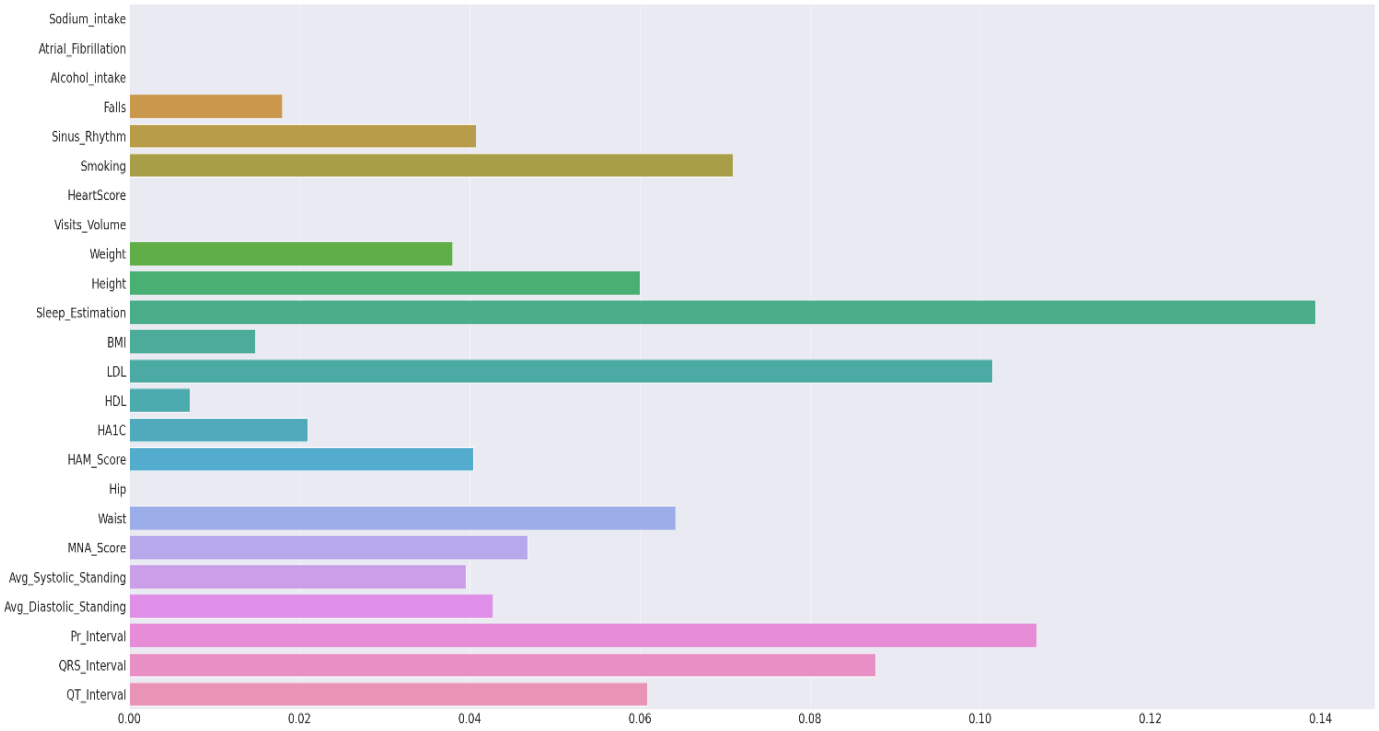


Figure 9: Features importance in the CVD prediction using the PoP dataset.

Cross-Validation Metrics Summary:

	mean	sd	cv_1_valid	cv_2_valid	cv_3_valid	cv_4_valid	cv_5_valid
accuracy	0.8470588	0.0892103	0.8235294	0.8823530	0.9411765	0.7058824	0.8823530
auc	0.7673485	0.1050002	0.8125	0.7666667	0.8666667	0.5909091	0.8
err	0.1529412	0.0892103	0.1764706	0.1176471	0.0588235	0.2941177	0.1176471
err_count	2.6	1.5165751	3.0	2.0	1.0	5.0	2.0
f0point5	0.5454902	0.1956731	0.2941177	0.5	0.8333333	0.6	0.5
f1	0.5545098	0.1277779	0.4	0.5	0.6666667	0.7058824	0.5
f2	0.6075396	0.1486931	0.625	0.5	0.5555556	0.8571429	0.5
lift_top_group	1.7	3.8013155	0.0	0.0	8.5	0.0	0.0
logloss	0.5203734	0.2995809	0.2885698	0.4398001	0.3804229	1.0439289	0.4491453
max_per_class_error	0.4284091	0.1361030	0.1875	0.5	0.5	0.4545455	0.5
mcc	0.5094937	0.1084346	0.4506939	0.4333333	0.6846532	0.5454546	0.4333333
mean_per_class_accuracy	0.7724621	0.0784699	0.90625	0.7166666	0.75	0.7727272	0.7166666
mean_per_class_error	0.2275379	0.0784699	0.09375	0.2833333	0.25	0.2272727	0.2833333
mse	0.1727154	0.0941231	0.1084299	0.1473326	0.1345058	0.3391809	0.1341277
pr_auc	0.3302299	0.1878055	0.1369538	0.2528322	0.6353569	0.3579570	0.2680497
precision	0.5590909	0.2725378	0.25	0.5	1.0	0.5454546	0.5
r2	-0.4901711	0.2745129	-0.9585144	-0.4193035	-0.2957396	-0.4852011	-0.2920969
recall	0.7	0.2738613	1.0	0.5	0.5	1.0	0.5
rmse	0.4057007	0.1007617	0.3292869	0.3838392	0.3667504	0.5823924	0.3662345
specificity	0.8449243	0.1805646	0.8125	0.9333333	1.0	0.5454546	0.9333333

Figure 10:The calculated metrics for the cross-validate data

2.4 Analytics for Hearing Loss

This section concerns the clinical use case of the Hearing Loss scenario in the SMART BEAR project. In the Hearing Loss (HLoss) clinical use case several parameters could be improved: 1) increase the hours of usage, 2) increase the patient satisfaction with the Hearing Aid (HAid), 3) decrease the risk of dropping out, and 4) reduce the number of visits to the Audiologists' office for HA fine tuning. This section reports the pipeline proposed for the first case – increasing the HAid usage. The proposed pipeline comprises pre-processing the data, hyper-tuning the hyper-parameters, training the model with the selected best set of hyper-parameters, testing the model, and finally explaining the model. The predictive model constructed with this pipeline is used to predict future HAid usage, and the explanation model is used to explain the prediction.

2.4.1 Data

One of SMART BEAR's precursor project is the Evotion project[13] – in it, almost 1,000 participants with varying degrees of HLoss were recruited, and each participant was supplied with a pair of EVOTION HAids that were connected to a smartphone by low-energy Bluetooth [11]. Real-time data that includes both smartphone and HAids data were logged by the smartphone every minute, similar to how Smart Bear will collect data from participants HAids. In this pipeline we have used the sampled EVOTION Synthetic Data (EvoSynth) with 399,500 observations of 53 participants [12]. EvoSynth is chosen since the variables are similar to what SMART BEAR collected during the PoP for the HLoss case. Since EvoSynth is an open-source dataset, no personalised features were included. However, it is considered necessary to include some personal characteristics in order to accomplish the objectives for this pipeline, which is to identify characteristics that encourage participants to use their HAids for a sufficient period of time during the day. Therefore, two additional variables that are regularly collected by clinicians during audiological clinical evaluations and also collected by SMART BEAR – Age and Sex – are randomly populated and added to the dataset. Table 7 summarises the variables with their description and type.

Table 7. Variables Names and Descriptions in the EvoSynth Dataset

Variable Name	Description	Type
ID	Identifier	Integer
Age	Years of age	Integer
Sex	Biological gender: female and male	Categorical
hProg	HAid program: low, medium, high, and high+	Categorical
hVol	HAid volume	Integer
LatRel	Relative latitudinal	Continuous

[13] https://h2020evotion.eu/

LonRel	Relative longitudinal	Continuous
PTA4	Pure tone average across 4 testing frequencies (0.5, 1, 2, and 4kHz)	Continuous
SoundClass	Sound environment: quiet, speech, speech in noise, and noise	Categorical
Timestamp	Time of the logged record	ISO 8601

Pre-processing steps include calculating the daily usage from the timestamps; a HAid is deemed as inactive when the difference between two consecutive timestamps is more than 600 seconds. Afterwards, the variables are transformed with encodings according to their type – so *hProg* and *Soundclass* are ordinal encoded, *Sex* is one-hot encoded, and the continuous variables are standardised and normalised. For other variables to also have a daily frequency, the average of each variable is taken for each day for each participant. The Trajectory Mean method is used to handle missing data, since it is suitable for longitudinal data, where the mean of the observed value per participant is imputed. For detecting and removing outliers, the standard deviation method is employed after the data is standardised and normalised. The number of standard deviations away from the mean, ζ_i, for every data point is calculated first. Then the z-score for every ζ_i is calculated to determine the outliers. If the z-score standard deviation of a data point is higher than a pre-defined threshold, then this data point can be considered to be an outlier. The threshold is set to three in this pipeline. Finally, the data is split into training, validation, and testing sets. In particular, the splitting takes place for each participant, where 80% of the data per participant is used as the training set, 20% of the training set is used as the validation set, and 20% of the data is reserved as the unseen testing set. All training, validation, and testing sets are then merged, so that the model can be trained with global data, yet still be able to make personalised predictions, as well as global predictions.

2.4.2 AI Model and Approach

The pipeline uses an attention-based Encoder-Decoder (attn-ED) system as the predictive model, for predicting participants' future daily HAid usage. The Shapley Additive exPlanations (SHAP) explanation method is then employed to identify characteristics that influence the model prediction. With the proposed pipeline, clinicians also have the choice to conduct either a personalised prediction/explanation or a global one.

2.4.2.1 Prediction Model

Figure 11. The architecture of the proposed model – attn-ED

As shown in the model architecture in Figure 11, the encoder (layer 2) uses an LSTM (Long Short-Term Memory networks) to encode the input sequences into a vector with fixed dimensionality, while the decoder uses another LSTM in layer 5 to decode the target sequences from the fixed vectors. The specific Attention Mechanism employed in the pipeline (layers 3) is similar to the self-attention mechanism proposed by Vaswani et al. [13] and takes the sequence of hidden state vectors for all timesteps produced by the encoder-LSTM as inputs. In self-attention, the representation of an input sequence is computed by relating different positions within the sequence. At each timestep, a compatibility score between the hidden state vector that the attention needs to be determined against all other hidden state vectors is calculated, such that,

$$Compatibility\ score\ = \frac{HH^{\top}}{\sqrt{d_H}}.$$

where d_H is the dimension of the sequence of hidden state vectors. Finally, the output of the attention mechanism is calculated as a weighted sum of the hidden state vectors and the compatibility score. The matrix of the output is calculated as follows:

$$Attention(H) = softmax\left(\frac{HH^{\top}}{\sqrt{d_H}}\right)H.$$

Table 8. Hyper-parameters Settings

Hyper-parameters	Settings
Number of LSTM hidden units	32, 64, 128, 216, 512
Dropout rate in the LSTM layer	0.01-0.0001
Recurrent dropout rate in the LSTM layer	0.01-0.0001
Dropout layer	0.01-0.0001
Activation function in the fully-connected layer	ReLu, Sigmoid, Tanh
Learning rate	0.01-0.0001
Batch size	16, 32, 64

To optimise the performance of the predictive model, the hyper-parameters listed in Table 2 are hyper-tuned first with the validation set. Then, the model is trained with selected best set of hyper-parameters with the training set for 500 epochs with early call-backs to avoid overfitting. To evaluate the performance of the model, three standard error estimators are used:

$$sMAPE = \frac{200}{N}\sum_{t=1}^{N}\frac{|y_i - \tilde{y}_i|}{|y_i| + |\tilde{y}_i|},$$

$$MAPE = \frac{1}{N}\sum_{t=1}^{N}\frac{|y_i - \tilde{y}_i|}{y_i},$$

$$WAPE = \frac{\sum_{i=1}^{N}|y_i - \tilde{y}_i|}{\sum_{i=1}^{N}|y_i|}.$$

where y_i is the true value, $\tilde{y}_i$ is the predicted value, and N is the number of data points.

The best set of hyper-parameters are listed in Table 3. In order to benchmark the predictive performance of attn-ED, its results are compared with a Vanilla LSTM, which contains just an input layer, an LSTM layer, and an output layer.

To obtain a personalised prediction, attn-ED is trained on all training sets but then tested on predicting for one specific participant. Results shown here are of Participant 17. Table 3 shows that attn-ED outperformed the Vanilla LSTM in all error estimators.

Table 9. Comparison of the Error Estimators Between attn-ED and Vanilla LSTM in the case of Personalised Prediction

Error Estimators	attn-ED	Vanilla LSTM
sMAPE	***0.2460***	1.2158
MAPE	***0.7576***	12.3022
WAPE	***0.2513***	1.0094

In the case of a global prediction, attn_ED is trained on all training sets and then the future HAid usage for all participants are predicted. Such that we are using all available historical HAid usage and other features for the entire population to predict future HAid usage, so that the features impacting future HAid usage for all participants can be found. Table 4 shows that attn-ED also outperformed the Vanilla LSTM in all three error estimators.

Table 10. Comparison of the Error Estimators Between attn-ED and Vanilla LSTM in the case of Global Prediction

Error Estimators	attn-ED	Vanilla LSTM
sMAPE	***0.1394***	0.4588
MAPE	***1.7247***	7.4789
WAPE	***0.1572***	0.4019

2.4.2.2 Explanation Model

SHAP, more specifically, kernel SHAP is used to explain the prediction results. Although SHAP is designed to be local explanation method (e.g., only one instance in the test set can be explained), it is however possible to obtain a global explanation (e.g., an explanation for all instances in the test set) from SHAP by calculating the mean absolute SHAP values for each feature across the dataset is taken. This way, the relative impact (e.g., the impact of one feature on the predictions with respects to all other features) of all features over the entire dataset and the global importance of each feature can be determined. SHAP results are visualized using the SHAP summary Plot, which combines feature effects and feature importance.

Figure 2 shows the SHAP explanation for the personalised prediction for participant 17. All features are listed in (decreasing) order according to their importance on the y-axis of the plot, and the x-axis represents the SHAP values, while the feature values (e.g., the actual values of a feature) from low to high are indicated by the colour gradient. The figure shows that *Usage* is the most important feature contributing to the prediction for Participant 17, with all other features making a negligible contribution to the prediction, such that their SHAP values are either zero or very close to zero. We can see that a higher daily HAids Usage (red values plotted on row Usage) corresponds to a higher future Usage (the SHAP values of the red values are positive), while a lower HAid Usage (the blue values plotted on row Usage) leads to a lower future Usage (the SHAP values of the blue values are negative). On the other hand, a lower value of PTA4 only marginally reduces what the model would have predicted as daily usage (e.g., only some of the blue values plotted on row PTA4 have small negative SHAP values).

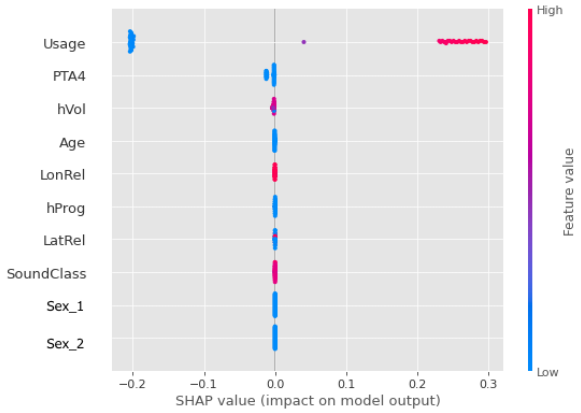


Figure 12. SHAP Visualisation in the case of Personalised Explanation

It is also observed that higher daily HAid usage leads to higher future usage among all participants, as shown in Figure 3. While all other features contribute very little to the model prediction, *hVol* is the second most important feature contributing the model prediction for all participants, and *SoundClass* is the least important feature. It is also worth to note that *Sex_1* corresponds to *Female* and *Sex_2* corresponds to *Male*. Therefore, Figure 3 also indicates that female participants contribute more to the model prediction than male participants.

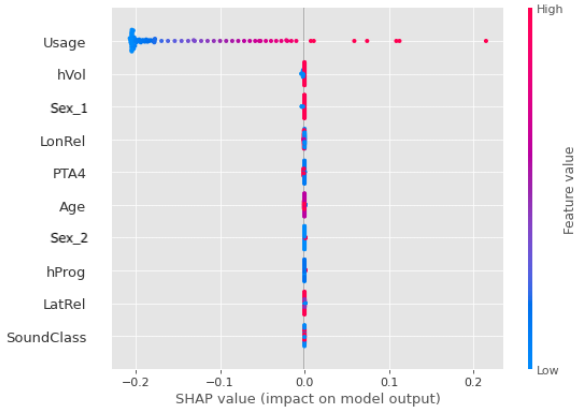


Figure 13. SHAP Visualisation in the case of Global Explanation

2.5 Federated Learning Framework

As described in Deliverable 4.2 Report on SMART BEAR Cloud Enabling Components v1, Federated learning is a machine learning technique that can train an algorithm across multiple systems or devices holding local datasets without the need of exchanging data. Instead of data, models and model parameters are exchanged. This approach is well suited as well for personalization of models to individual users, in a network of IoT devices and smart devices such as mobile phones or dedicated edge devices. Even when no geographic distribution of data is necessary, this architecture enables the separation of local datasets for training and supports the implementation of increased autonomy with respect to local model updates and contribution to the global model between the coordinator and the individual worker nodes.

The data management scenario in SMART BEAR is centralized, all data being collected in the repository hosted on the cloud. Therefore, we do not leverage the privacy and distribution benefits of the federated learning approach. The key benefit is in the ability to personalize to individuals and to their data. In SMART BEAR we develop techniques to tailor the federated learning solution towards individual user-level personalization.

Specific extensions are required to assess the performance of the local and global models, to implement validation loops to help detect any performance degradation, to identify data outliers and make decisions on their use in the system, and to design patterns of data and parameters exchange between the workers responsible for the local models specific to individual patients and the coordinator responsible for the global model. This solution facilitates as well future scenarios in SMART BEAR, ensuring scalability when data from external systems becomes available for model development without the need to be transformed and loaded into the SMART BEAR repositories, or when additional cloud resources need to be leveraged for data storage or for computation.

2.6 AI models and interventions in the CVD scenario

2.6.1 Models

There is an extensive number of different models and networks that one can use to create predictions [10]. Choosing the best suited model for a use case depends on multiple factors such as type of input, predictive strength, and explainability. For the cardiovascular disease (CVD) deliverables we plan on using two advanced machine learning models, LSTMs who have great predictive strength for time series input, and Neural Tree models which are easier to interpret.

2.6.1.1 LSTM

LSTM (Long Short-Term Memory see figure 1) is a type of neural network that is designed to handle sequences of data, such as text, speech or continuous vitals [8]. Unlike traditional neural networks that can struggle to remember previous information, LSTMs have "memory cells" that allow them to store important information from earlier in the sequence. Additionally, LSTMs have "gate units" that regulate the flow of information into and out of the memory cells, enabling the network to selectively remember and forget information as needed. This makes LSTMs well-suited for tasks where context and past information are crucial factors. By incorporating these memory and control mechanisms, LSTMs are able to produce more accurate predictions than traditional neural networks in many cases.

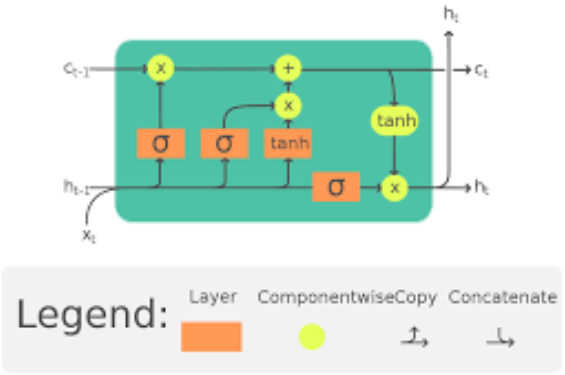
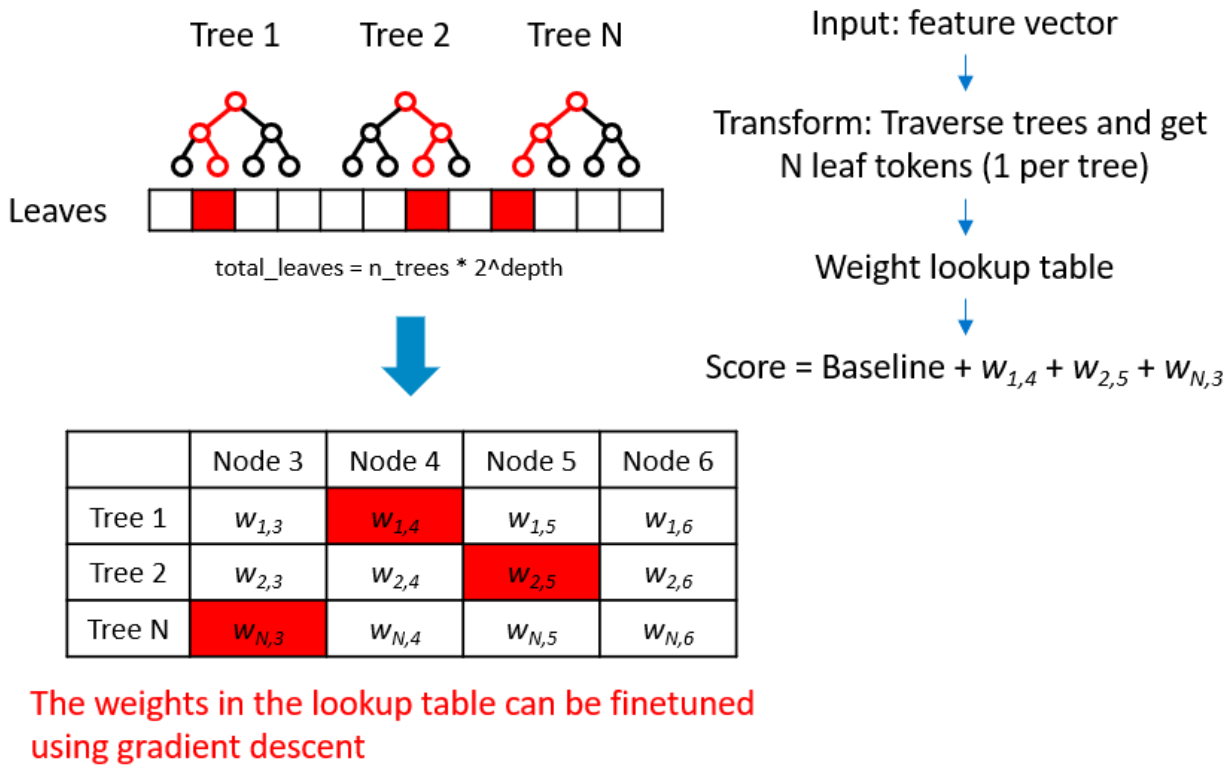


Figure 14 LSTM Cell Architecture [14]

[14] https://upload.wikimedia.org/wikipedia/commons/thumb/9/93/LSTM_Cell.svg/450px-LSTM_Cell.svg.png

2.6.1.2 Neural Tree model

Neural tree models are a type of deep learning architecture that combines the strengths of decision trees and neural networks [9] (see Figure 15). They are used to model hierarchical structures, where the input data is processed through a series of nodes or "decision points", similar to a decision tree. However, unlike decision trees, the nodes in a neural tree model are represented by artificial neural networks, which allows for more complex decision-making and non-linear relationships between inputs and outputs. Neural tree models are used in various applications such as natural language processing, image classification, and drug discovery, where the input data can be decomposed into a tree-like structure with multiple levels of abstraction. They offer an effective way of combining the interpretability and structure of decision trees with the power of neural networks. We have successfully applied this architecture in [14].



	Node 3	Node 4	Node 5	Node 6
Tree 1	$w_{1,3}$	$w_{1,4}$	$w_{1,5}$	$w_{1,6}$
Tree 2	$w_{2,3}$	$w_{2,4}$	$w_{2,5}$	$w_{2,6}$
Tree N	$w_{N,3}$	$w_{N,4}$	$w_{N,5}$	$w_{N,6}$

Figure 15 Neural Tree Model

2.6.2 Applications of these models to support specific use cases in SMART BEAR

2.6.2.1 Heart rate prediction during exercises

High heart rates have shown to be a risk factor for patients with a cardiovascular disease [15]. Furthermore, research has shown that planning the optimal exercise using heart rates can lead to better health outcomes [16]. By monitoring heart rate during exercise and adjusting the intensity based on set targets, individuals can improve their health more efficiently and safely. LSTM models can be applied in this field by combining heart rate data with other relevant information, such as age but not limited to, continuous heart rates during exercises, weight, history of previous exercises and medical history, to create personalized exercise plans and predict the optimal

heart rate zones for different individuals. This can help improve the effectiveness and efficiency of exercise programs, leading to better health outcomes.

2.6.2.2 Sleep duration prediction

Sleep is an essential component of overall health and well-being. It plays a crucial role in repairing and rejuvenating the body, regulating hormones, and consolidating memories. Inadequate or poor quality sleep can have negative impacts on physical and mental health, including weakened immunity, decreased cognitive function, and increased risk for chronic conditions such as obesity, diabetes, and cardiovascular disease. For patients, the importance of adequate and restful sleep cannot be overstated. By prioritizing sleep and addressing any sleep-related issues, patients can support their overall health and well-being. Neural tree models can be trained to predict the hours of sleep. When we can identify in each patient which variables are most important in predicting sleep time, recommendations can be provided if for example a patient should not exercise in the evening or should take a walk before going to bed etc. For this model to be trained we need data such as but not limited to, continuous heart rates, history of hours of sleep, age, weight, and medical history.

2.6.2.3 Deterioration prediction and Regimen adherence

Regimen adherence is a critical component in the successful management of a patient's health [17]. When a patient adheres to their prescribed treatment plan, it helps to ensure that they receive the full benefit of their medication, leading to improved health outcomes. Furthermore, regimen adherence can reduce the risk of adverse events and deterioration.

Neural Tree models can be used to predict if a patient is going to deteriorate or if a patient is adhering to the prescribed regimen. Continuous heart rates can be used as a proxy for labelling deteriorations and disobedience of the described regimen. Patient specific information such as but not limited to, continuous heart rates, history of hours of sleep, age, weight, and medical history is needed to create a well performing model.

2.6.3 Data

To train such models data with all relevant data elements and of good quality is needed. In order to predict the heart rate during an exercise (semi) continuous HR measurements are needed as well as the time at which an exercise is performed. Furthermore, other variables such as weight, drinking status, smoking status, hours of sleep, etc. could be used to increase the performance of such models. We have explored the data collected in the PoP to ensure that all the elements required for these models are collected at the needed level of detail and accuracy.

2.6.3.1 Current challenges

Data such as hours of sleep, smoke status and weight are already available in the data set. Even though a lot is already available there are still some issues we are trying to resolve. First, the overall quality of the data can be improved, problems are mainly related to the observations table, below are some examples that were found but the problems are not limited to those. For example, see Figure 16 which shows the walking distance of a patient over a period of 24 hours. As you can see there are multiple entries at the exact same data for the same patient with different values. The correct value should be the largest one as the other values are measured during the day, what should be fixed here is the timestamp of the other values. It is quite valuable information when a patient took a walk, definitely in combination with the heartrate.

	patient_id	effectiveDateTime	valueQuantity_value	code_coding_code	valueCodeableConcept_coding_code	valueCodeableConcept_coding_display	code_coding_display
4417	792607756	2021-11-26 01:00:00	11.466	41953-1	None	None	Walking distance 24 hour Calculated
4418	792607756	2021-11-26 01:00:00	4.194	41953-1	None	None	Walking distance 24 hour Calculated
4419	792607756	2021-11-26 01:00:00	1.842	41953-1	None	None	Walking distance 24 hour Calculated
4420	792607756	2021-11-26 01:00:00	4.088	41953-1	None	None	Walking distance 24 hour Calculated
4425	792607756	2021-11-27 01:00:00	5.726	41953-1	None	None	Walking distance 24 hour Calculated
4423	792607756	2021-11-27 01:00:00	0.038	41953-1	None	None	Walking distance 24 hour Calculated
4424	792607756	2021-11-27 01:00:00	1.611	41953-1	None	None	Walking distance 24 hour Calculated

Figure 16 Walking distance in the dataset

Secondly, Figure 17 shows aggregated heart rate values over the last 24 hours. As can be seen in the figure is that again there are multiple entries on the same exact moment. For maximum and minimum this is not a big problem as we can take the maximum of the Max and the minimum of the Min but for the average there is no way of knowing what the actual average is.

473	32578327	2022-01-14 01:00:00.000	137.000	55426-1	None	None	Max heart rate
552	32578327	2022-01-14 01:00:00.000	128.000	55426-1	None	None	Max heart rate
798	32578327	2022-01-14 01:00:00.000	70.000	55425-3	None	None	Average Heart rate
1060	32578327	2022-01-14 01:00:00.000	71.000	55425-3	None	None	Average Heart rate
1100	32578327	2022-01-14 01:00:00.000	86.000	55425-3	None	None	Average Heart rate
1660	32578327	2022-01-14 01:00:00.000	53.000	96935-2	None	None	Min heart rate
1724	32578327	2022-01-14 01:00:00.000	64.000	96935-2	None	None	Min heart rate
1843	32578327	2022-01-14 01:00:00.000	48.000	96935-2	None	None	Min heart rate

Figure 17 Aggregated heart rate values

Additionally, the observations table is populated with an extreme amount of NaN values, in the observations table there are 67 columns of which on average 53 are filled with NaN values.

Finally, for the advanced machine learning models semi continuous heart rate data is needed which is not yet available in the data set but is measured by the wearable. Without these variables the previously described models cannot be trained.

2.6.3.2 Possible solutions

A few simple steps can be taken to improve the quality of the data. First, providing the exact moment of when a patient was walking will solve the problem of multiple entries. Secondly, only adding the correct aggregated heart rate values in the data set will also improve the data quality. Additionally, taking a look at what columns are needed and not needed and removing the columns with only NaN values will also improve the quality of the data. Finally, and most importantly, the continuous heart rate values are collected, but not sent to the FHIR-based repository. So, by either creating a FHIR resource and extending the repository or saving the values in a different kind of format in another repository so they can be used for analytics is of the highest priority.

Furthermore, in order to collect the walked distance and speed we can collect the differences in locations between two time points (close together). This information is needed to better correlate heart rate values with exercising.

2.6.3.3 Chosen solutions

After discussing with the team, we decided to store the semi-continuous heart rates (as recorded from the devices) in a non-FHIR repository as it is not relevant for the dashboard. For advanced analytics the data is not needed to be accessed in real time and the database can iteratively be built up. Therefore, storing it in a non-FHIR repository where it can be retrieved in an on-demand fashion was chosen as the most efficient solution.

2.7 Dashboard

The SMART BEAR dashboard is a component aimed at providing a user-friendly graphical user interface for the clinicians of the platform. The interface allows the clinicians to register and see a patient, his/her questionnaire score and his/her medical history, in addition to the delivered interventions. This section describes the main functionalities provided by the dashboard in order to create a patient, register the results of his/her physical examination, assessment, and the delivered interventions. The functionality that are shown here support the PoP, and the implementation of interaction elements continues on the basis of the requirements elaborated in D2.1.

The dashboard is the component with which the end user interacts most of the time. Depending on the role assigned to the user, the functions change and also the pages with which it is possible to interact are differentiated. For example, the SysAdmin type user cannot view the clinical data, while the clinician cannot manage users within the system. The TechSupport type user can view patient data, but can monitor the sending of data and notify managers to check that there are no communication problems that could affect the sending of data in the system.

The user who has the most functions at his disposal is certainly the clinician, he can not only manage user registration, enter the data obtained during an assessment, but also launch analytics and check the data sent within the platform.

2.7.1 Registration and login

Before using the dashboard any type of user must register through the form that is exposed by WSO2 which is the component that manages the authentication mechanism. The system requires entering a nickname and then waiting for an activation email in which the user will receive a link to use. After the activation, the user can login (see Figure 18 and Figure 19). In the meantime, one of the system administrators will take care of assigning Role, Organization, and activating the user. The System administrator ecan see all the list of users that needs an approval and then decide which one to accept or reject and also modify the already existing ones in the platform (see Figure 20).

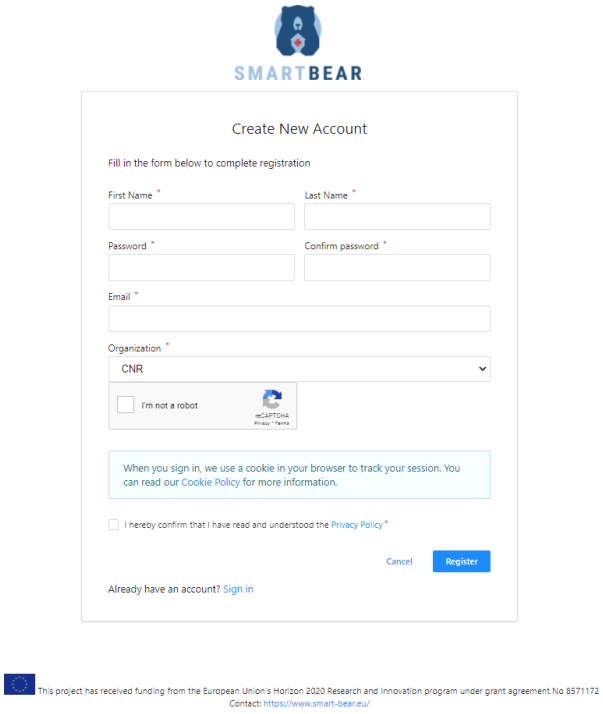


Figure 18. The registration form

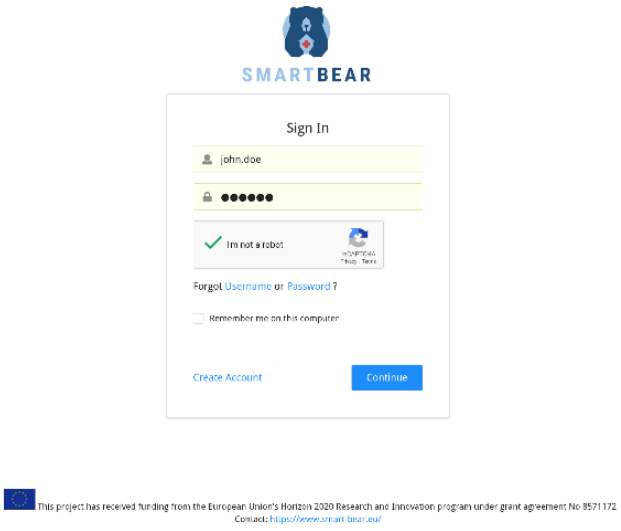


Figure 19. Snapshot of signing page to access the Registration and Logging page

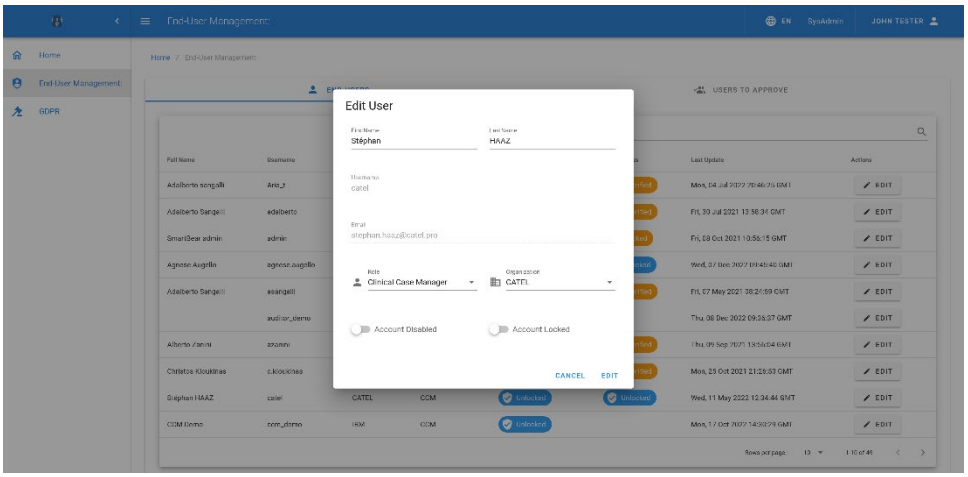


Figure 20. End user management page which enables the editing

2.7.2 Patient creation

The creation of the patient is a fundamental step that can only be performed by the clinician. To do this, the dashboard uses the User Management component, this is made necessary because the User Manager will manage all sensitive data and is the only component to have the mapping between the ids between the data in the FHIR repository and those in its internal. Through the form it is necessary to specify some data such as the telephone number, in case it is necessary to contact the user again, the email and the CCM that will follow him (see Figure 21).

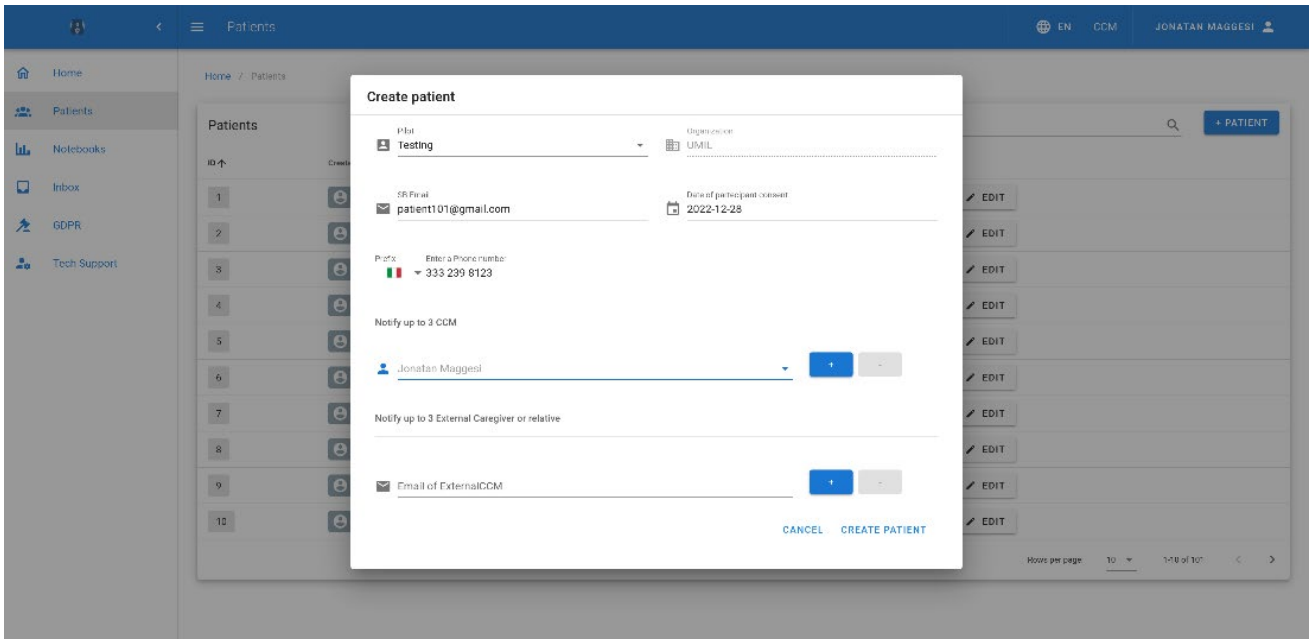


Figure 21. The form for creating a patient

2.7.3 Notebooks and analytics

To launch analyses, the clinician has the section containing the notebooks available so that he can use the catalogue of analytics that are available in the system and configure them in order to launch the execution and obtain the results (see Figure 22 Figure 23 and Figure 24).

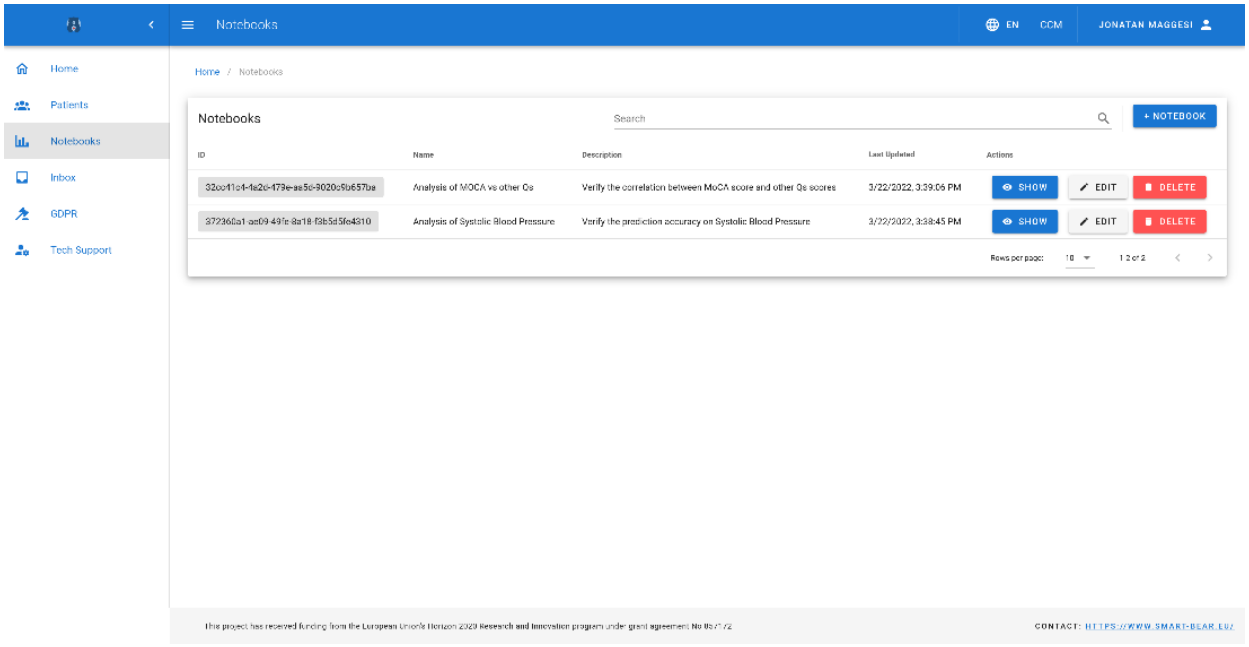


Figure 22. List of added Notebooks designed and launched by data scientists

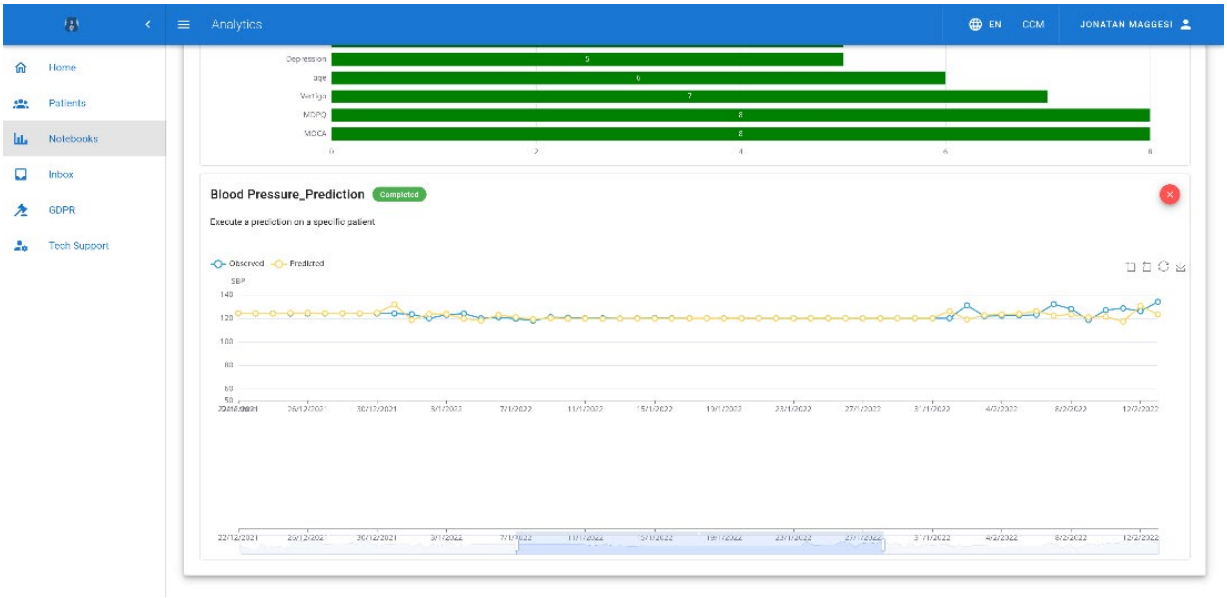


Figure 23. A closer look at the Blood pressure prediction notebook.

Figure 24. The creation of analytics

2.7.4 Assessments

In this section of the UI, the clinician enters the data that will be mapped into FHIR-type resources and then saved in the repository via REST calls (see Figure 25). Obviously, also in this case each call passes through the Security Component which takes care of translating the ID and avoiding having a link between sensitive data and the patient's ID. Based on the data entered in the first assessment, a profile is suggested for each patient, but it is always the clinician who decides on the basis of his/her experience how to profile the patient. During the first assessment it is also necessary to carry out a pre-test, in order to verify if the patient meets the requirements of the protocol and for this, they must pass a MoCA Test and show that they have a certain level of Dexterity.

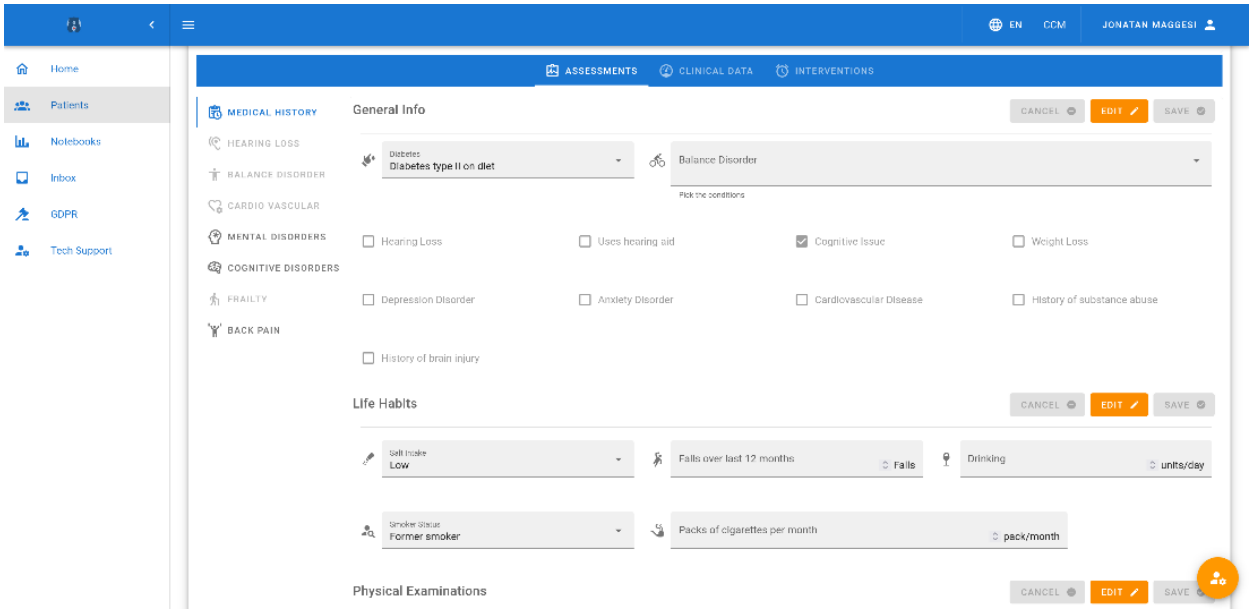


Figure 25. Medical History of a patient. This information is stored by clinicians during the assessments.

The list of questionnaires to be administered also changes based on the patient's profile. The UI shows them both in the Medical Assessment part and divided by profiling. The score obtained by the patient is always shown as a recap, but it is always possible to open one of the questionnaires, both to modify it and to check the answers entered by the patient (see Figure 26 and Figure 27).

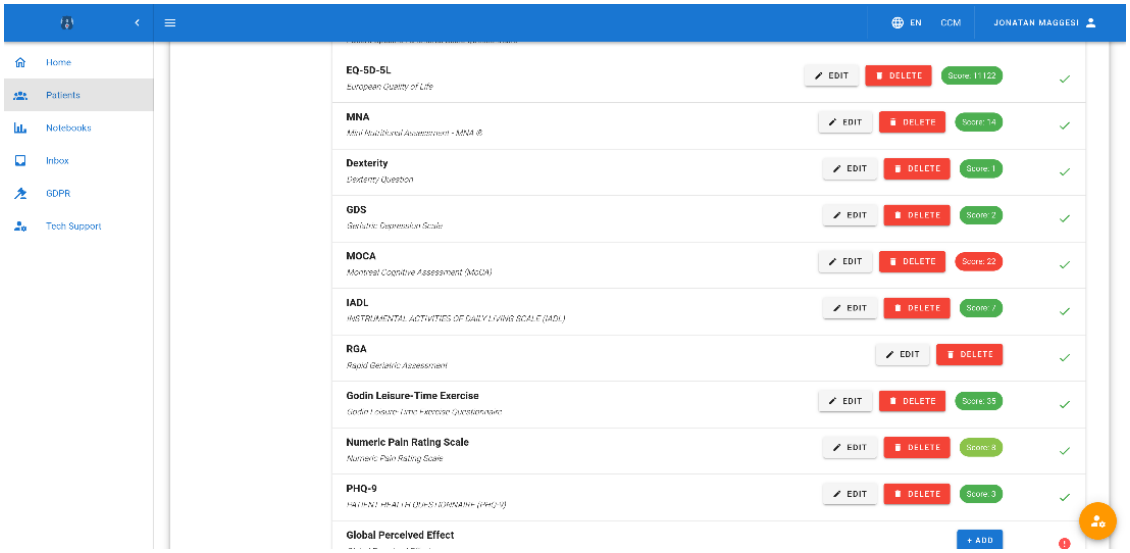


Figure 26. List of questionnaires and scores for the patients under study. The edit and delete buttons are also active for modification by relevant clinician.

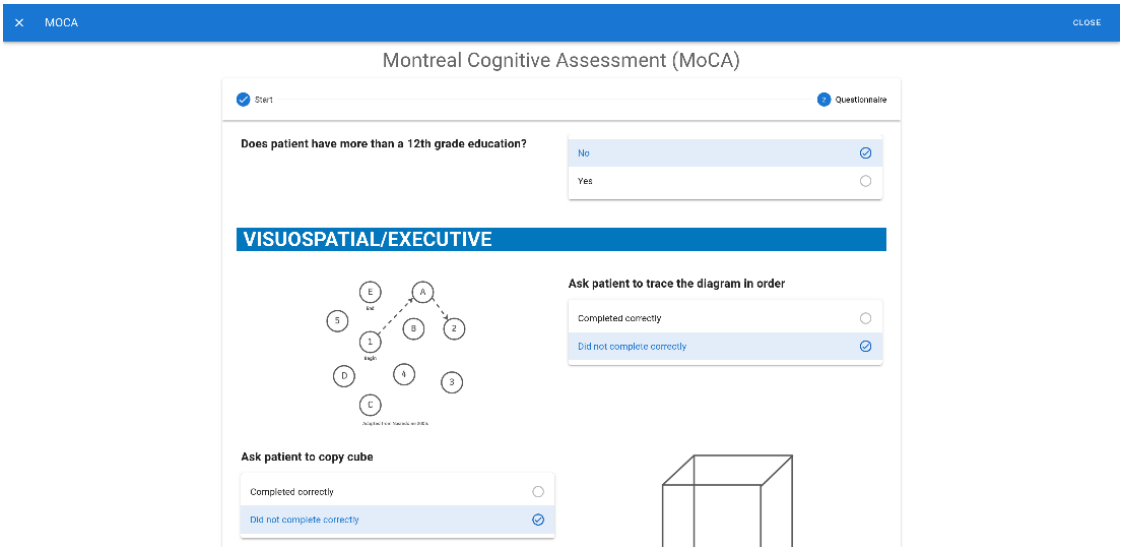


Figure 27. A view of the MoCA questionnaire used in the first assessment

2.7.5 Device allocation

The devices area and the external ids area on the patient page are the two tabs dedicated respectively to entering pairings with external tools such as smartphones and ids related to synergies with other projects. In the device section, the clinician enters the pairing with the devices assigned to the patient, these change according to the profile assigned to the patient and it is also possible to manage the pairing status. So through the UI it is possible to activate the pairing or disable it. (see Figure 28 and Figure 29) The same thing can also be done in the area that deals with pairing with project ids such as Holobalance[15] or Smart4Health[16].

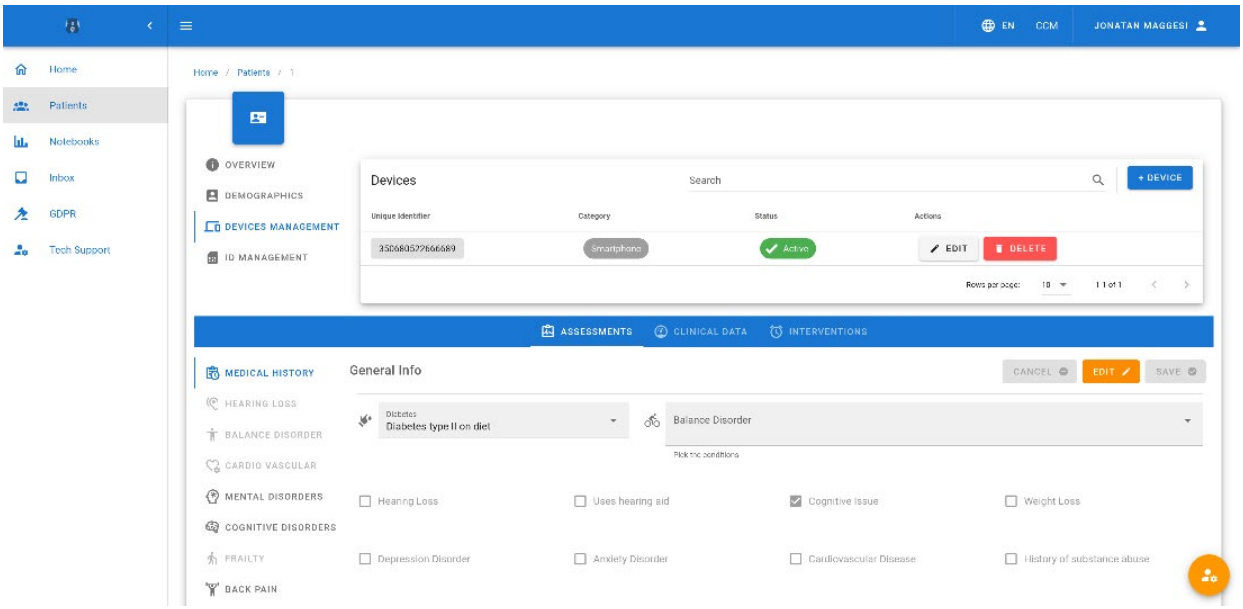


Figure 28. Information on the devices assigned to a patient

[15] https://holobalance.eu/

[16] https://smart4health.eu/

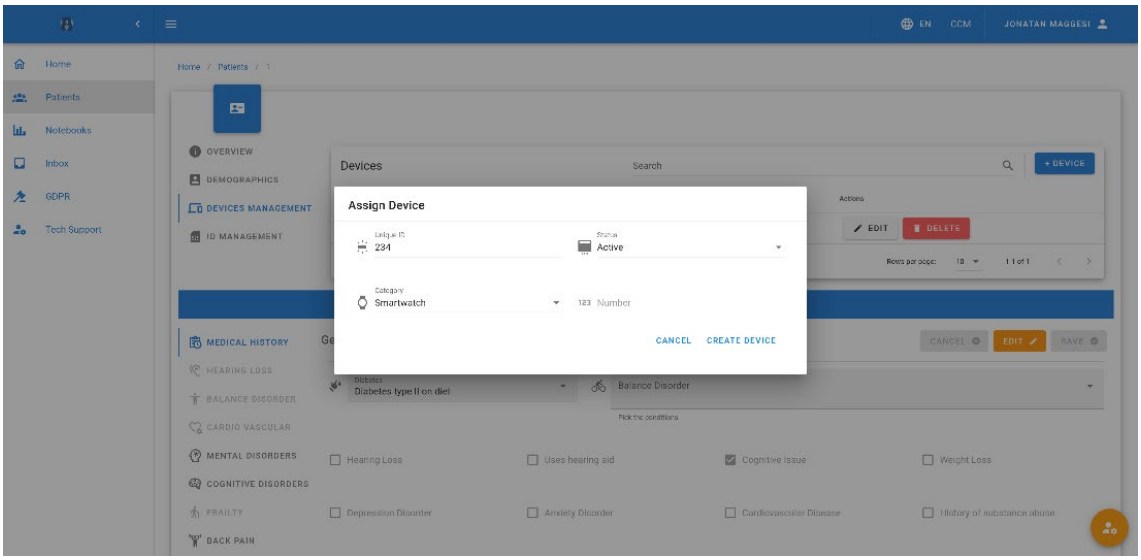


Figure 29. Device assignment page

2.7.6 Tech support

This part of the UI is only visible to clinicians and Tech Support users. This was specifically created to check that the data arrives on the platform and, if any anomaly is noted, contact the patient to try to understand the reason for not sending the data. As shown in the pictures the user can see a list of patients and the days of inactivity. As shown in the figures, the user can see a list of patients and the days of inactivity (see Figure 30). By clicking on the button, it is possible to: enter the overview interface for a patient, see the pairings, when the last transmission took place and possibly also open the log files including the list of transmissions to check if some of these are not valid and try to understand where the problem has occurred (see Figure 31 and Figure 32).

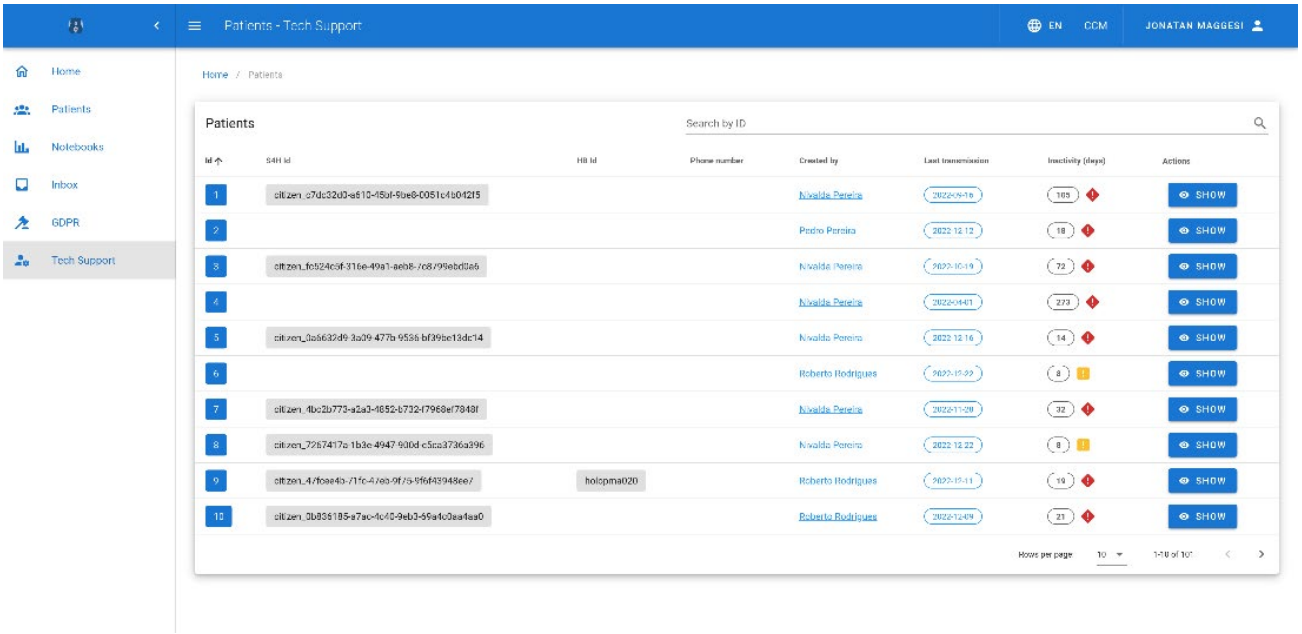


Figure 30. Technical support page to be managed by technicians.

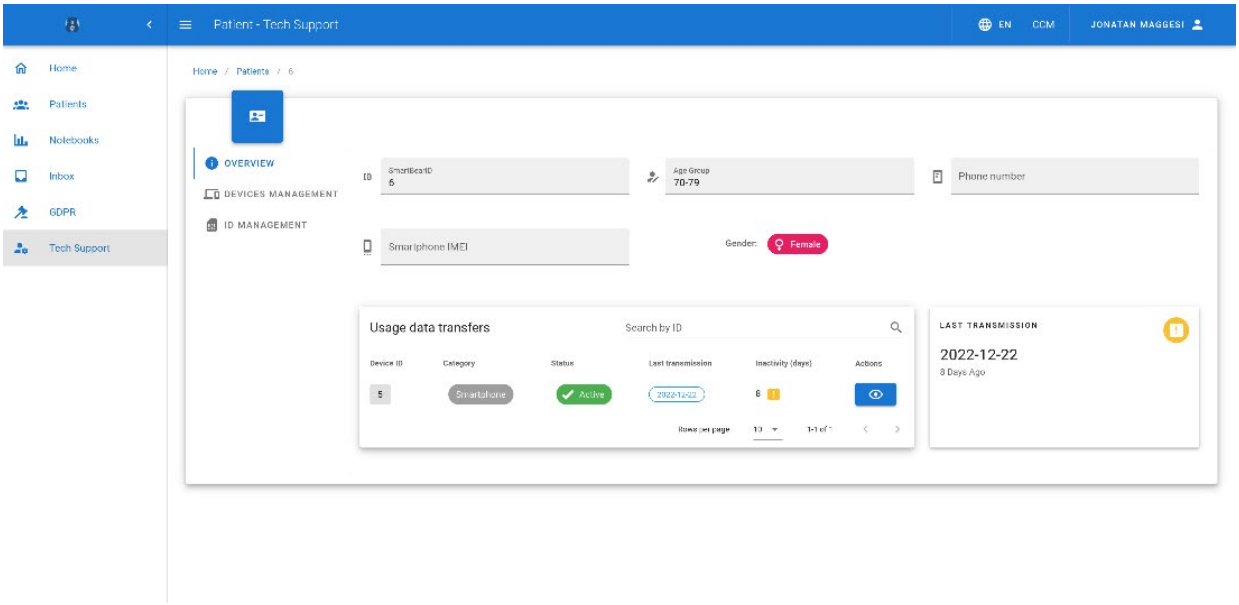


Figure 31. A patient's overview page for the technicians

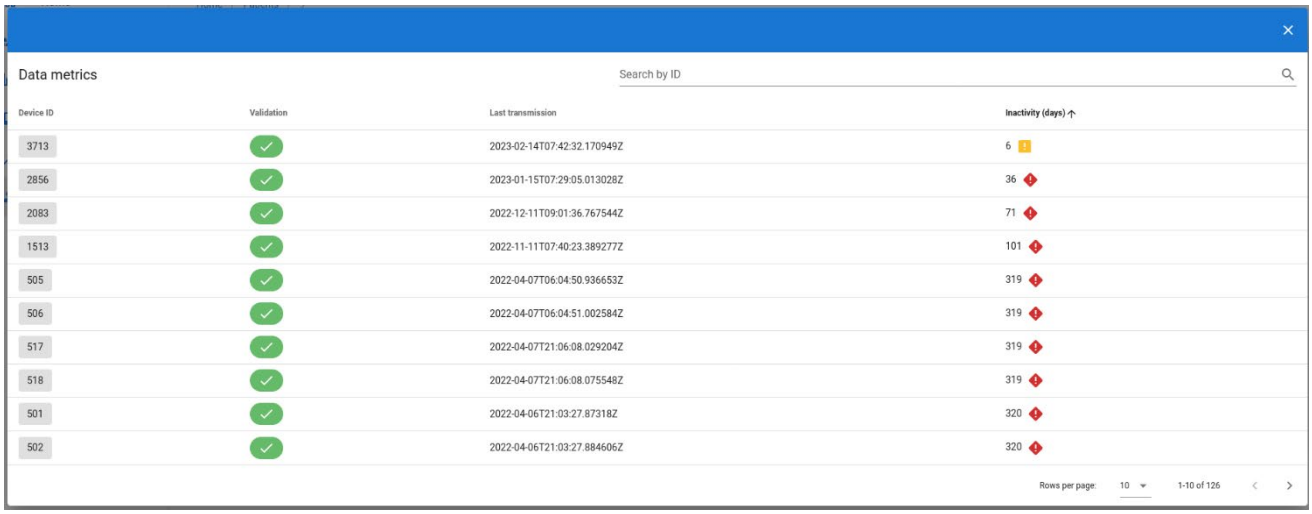


Figure 32. A screenshot showing the log files

3 Concluding Remarks

This deliverable provides the specification of the refined versions of the relevant SMART BEAR infrastructure components that compose the central cloud platform, including the data model, the implementation guide, the Big Data Engine and the analytics components.

The FHIR data model and the Implementation Guide were extended to support additional data for the hearing loss use case.

We present as well the first analytics results with SMART BEAR data, the analysis we carried out for the data collected and its ability to support the desired interventions described in the clinical use cases. For some planned analytics we identified issues with the data for which we have implemented a solution that will enable us to develop those models with the relevant data. Future reporting will include the results of those models evaluated with SMART BEAR datasets.

4 References

[1] P. Katrakazas, T. Androutsou, A. Smith, J. Milas, D. Brdarić, L. Murdin, N. Nina Koloutsou, A. Dudarewicz, M. Pawlaczyk-Łuszczyńska, M. Śliwińska-Kowalska, N. H. Pontoppidan, L. Trenkova, G. Kaloyanova, N. Tsokova, D.-E. Bamiou, G. Dritsakis, A. Bibas and D. Kikidis, "Public Health Policy Decision Models (PHPDM) v1, Deliverable D3.1 to the EVOTION-727521 Project funded by the European Union," ICCS, Greece, 2017.

[2] P. Katrakazas, D. Koutsouris, K. Koloutsou, A. Smith, G. Spanoudakis, K. Pozdniakov, L. Murdin, M. Sladen, D. Brdaric, S. Samardžić, I. M. Huskić, M. Kowalska, J. H. Christensen, L. Trenkova, D.-E. Bamiou, G. Dritsakis, M. Anisetti, A. Bibas and D. Kikkidis, "Public Health Policy Decision Models (PHPDM) v2, Deliverable D3.2 to the EVOTION-727521 Project funded by the European Union," ICCS, Greece, 2019.

[3] I. Basdekis, Y. Bin, G. Spanoudakis, N. Dimakopoulos, P. Kanelopoulos, P. Kokkinakis, M. Smyrlis, P. Katrakazas, N. H. Pontoppidan and L. H. Tietz, "Data Repository and Collection Components, Deliverable D5.2 to the EVOTION-727521 Project funded by the European Union," City, University of London, London, 2017.

[4] W. Zucchini, I. L. MacDonald and R. Langrock, Hidden Markov models for time series: an introduction using R, CRC Press, 2017.

[5] I. L. MacDonald and W. Zucchini, Hidden Markov and Other Models for Discrete-valued Time Series, London: Chapman and Hall, 1997.

[6] L. R. Rabiner, "A tutorial on hidden Markov models," *In IEEE,* vol. 77, no. 2, pp. 257-286, 1989.

[7] G. D. Forney, "The Viterbi algorithm,," *In IEEE,* vol. 61, pp. 268-278, 1973.

[8] Sherstinsky, A. (2020). Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Physica D: Nonlinear Phenomena*, *404*, 132306.

[9] Chen, Yuehui, et al (2005). "Time-series forecasting using flexible neural tree model." *Information sciences* 174.3-4: 219-235.

[10] Raschka, S. (2018). Model evaluation, model selection, and algorithm selection in machine learning. *arXiv preprint arXiv:1811.12808*.

[11] J. H. Christensen et al., "Fully Synthetic Longitudinal Real-World Data From Hearing Aid Wearers for Public Health Policy Modeling," Zenodo https://doi.org/10.5281/zenodo.2668210. 2019.

[12] J. H. Christensen et al., "Fully Synthetic Longitudinal Real-World Data From Hearing Aid Wearers for Public Health Policy Modeling," Front Neurosci, vol. 13, no. 850, 2019.

[13] A. Vaswani et al., "Attention Is All You Need," Jun. 2017.

[14] Francesca Manni, Aleksandr Bukharev, Anshul Jain, Shiva Moorthy, Asif Rahman, Anca Bucur: Neural gradient boosting in federated learning for hemodynamic instability prediction: towards a distributed and scalable deep learning-based solution. AMIA 2022

[15] Perret-Guillaume, C., Joly, L., & Benetos, A. (2009). Heart rate as a risk factor for cardiovascular disease. *Progress in cardiovascular diseases*, *52*(1), 6-10.

[16] Karvonen, J., & Vuorimaa, T. (1988). Heart rate and exercise intensity during sports activities: practical application. *Sports medicine*, *5*, 303-311.

[17] Ingersoll, K. S., & Cohen, J. (2008). The impact of medication regimen factors on adherence to chronic treatment: a review of literature. Journal of behavioral medicine, 31, 213-224.

[18] Hughes, R.A.; Heron, J.; Sterne, J.A.; Tilling, K. Accounting for missing data in statistical analyses: multiple imputation is not always the answer. International journal of epidemiology 2019, 48, 1294–1304.

[19] Madley-Dowd, P.; Hughes, R.; Tilling, K.; Heron, J. The proportion of missing data should not be used to guide decisions on multiple imputation. Journal of clinical epidemiology 2019, 110, 63–73.