



Persistência de Dados e-Health num Sistema de Gestão de Ensaaios Clínicos

MIGUEL BRUNO PAIVA E SILVA

Junho de 2023

e-Health Data Persistence in a Clinical Trial Management System

Miguel Silva

**A dissertation submitted in partial fulfillment of
the requirements for the degree of Master of Science,
Specialisation Area of Software Engineering**

Supervisor: Dr. Ângelo Manuel Rego e Silva Martins

Evaluation Committee:

President:

Dr. Goreti Marreiros, Professor, DEI/ISEP

Members:

Dr. Paulo Oliveira, Professor, DEI/ISEP

Porto, June 29, 2023

Statement of Integrity

I hereby declare having conducted this academic work with integrity.

I have not plagiarised or applied any form of undue use of information or falsification of results along the process leading to its elaboration.

Therefore the work presented in this document is original and authored by me, having not previously been used for any other end.

I further declare that I have fully acknowledged the Code of Ethical Conduct of P.PORTO.

ISEP, Porto, June 29, 2023

A handwritten signature in black ink, consisting of a stylized, cursive script that appears to be the initials 'M.S.' followed by a large, sweeping flourish.

Abstract

The present thesis details the development of a platform belonging to a project focused on the development and commercialization of Smart Health solutions in Portugal. The platform in question aims to facilitate the integration testing and clinical trial of medical devices, utilizing a standard to establish a uniform set of semantics and nomenclatures and provide proper protection and storage of the data. For the development of the project, research was made into the matters of clinical trials, regulatory norms and standards capable of applying them, with HL7 FHIR being the clear standout. Furthermore, a value analysis of the idea of the platform was conducted, highlighting the interest for it and the benefits it might bring. This was followed by the proper analysis and design of the platform, which entailed not only documenting the context under which it would operate (its architecture, through multiple views), but also the various concepts and functionalities that were to be considered (its use cases and domain). The implementation of these ideas was also exposed, with the support of code snippets, and its consequent experimentation and evaluation was done to confirm its capacity in data storage and the efficiency in its performance when faced with rigorous data sets.

Keywords: Standards, FHIR, Clinical, Resource, Trials, Patients

Resumo

A presente tese está focada no desenvolvimento de uma plataforma de persistência e gestão de dados clínicos. Este projeto enquadra-se na iniciativa denominada SMART-HEALTH-4-ALL, que visa promover a conceptualização, desenvolvimento e comercialização de tecnologias Smart Health em Portugal. A plataforma em questão vai servir como ferramenta integral de suporte, no contexto de ensaios clínicos. Estes ensaios envolvem a troca de informação entre vários aparelhos Smart Health e outras entidades que vão executar revisões estatísticas, de forma a validar o correto funcionamento dos aparelhos e permitir que estes sejam aprovados para desenvolvimento em massa. Um dos maiores obstáculos enfrentados durante os ensaios clínicos é elevada disparidade na forma como os dados clínicos estão estruturados e designados nos aparelhos e sistemas envolvidos (tanto os que se sujeitam ao ensaio, como os que realizam as provas desse ensaio), levando ao uso de uma grande quantidade de tempo e recursos para uniformizar estas estruturas e possibilitar a correta execução dos ensaios. A plataforma proposta tem como objetivo facilitar e acelerar esta etapa no desenvolvimento de tecnologias médicas, propiciando o envio e receção dos dados clínicos, servindo como localização singular onde estas trocas de dados ocorrerão. De forma a prevenir a referida inconsistência nas definições e estruturações dos dados clínicos, a plataforma terá também a responsabilidade de fornecer uma definição uniforme de semânticas e nomenclaturas para a estruturação dos dados clínicos, promovendo a proteção e correta gestão dos mesmos. Para o desenvolvimento desta, investigações foram elaboradas, através de revisões de literatura, de forma a definir um estado de arte sobre os tópicos de ensaios clínicos (em que consistem e qual a sua importância e impacto no desenvolvimento tecnológico clínico), normas regulatórias (legais e de ética, que devem ser respeitadas e suportadas por ensaios clínicos e tecnologias usadas nas suas execuções), sobre gestão de dados clínicos e sobre standards de estruturação de dados capazes de respeitar essas normas e serem usados na implementação da plataforma. O estado de arte contém em particular uma análise significativa de uma série de standards, considerando o seu contexto histórico, as suas vantagens e desvantagens e a compatibilidade com a plataforma a desenvolver, com o standard denominado HL7 FHIR sendo o que mais se destacou e que provou ser o mais adequado para esta situação particular. Consequentemente, foi executada uma análise de valor à ideia da plataforma. Esta etapa envolveu uma definição mais concreta da ideia e das oportunidades que levaram à criação da ideia em si, através de pesquisas bibliográficas sobre testemunhos e estatísticas relevantes. Tendo isto, o valor da ideia foi determinado através do interesse por detrás da ideia e os vários benefícios que esta poderia trazer para as entidades que desfrutassem dela. De seguida, foram elaboradas as fases de análise e desenho de engenharia da plataforma. Iniciou-se documentando os vários conceitos e funcionalidades que iriam compor a plataforma em si, através de diagramas de domínio e da definição dos seus casos de uso, respetivamente. O desenho envolveu o planeamento e previsão dos contextos em que a plataforma se iria encaixar, com que outros componentes iria comunicar e como a transferência de informação ocorreria ao longo das suas camadas, ou seja, a potencial arquitetura do projeto, conseguida através de diagramas de sequência,

físicos e de componentes, que ilustravam as várias vistas a considerar. Tendo esta documentação efetuada e a análise e desenho concluídas, procedeu-se à exposição da implementação da plataforma em si, que se iniciou com a exploração de quais tecnologias usar (linguagem Java, framework de Spring e especificações de HL7 FHIR), procedendo posteriormente à detalhada série de passos para a configuração dos dois componentes chave da plataforma: o servidor HAPI FHIR e a API de gestão de dados. Para ambos, os passos a tomar para os preparar e executar foram apresentados com o auxílio de extratos de código e explicações dos mesmos, de forma a que qualquer indivíduo, mesmo que com conhecimento limitado de codificação, pudesse acompanhar o processo. Por conseguinte, tendo a plataforma completamente implementada, realizou-se a fase de experimentação e avaliação da solução. Para tal, determinaram-se inicialmente os indicadores (os aspetos da plataforma que iam ser avaliados). O primeiro destes indicadores foi descrito como a capacidade de corretamente persistir e gerir os dados clínicos. O segundo e terceiro indicador relacionaram-se com a performance da plataforma quando exposta a uma série consecutiva de dados clínicos e a agrupados (bundles) destes dados de grande dimensão, respetivamente. Para cada indicador foram estabelecidas duas hipóteses, uma com perspetiva positiva perante os resultados obtidos e outra com perspetiva negativa. Com isto, executaram-se as experiências através de simulações de pedidos HTTP à plataforma e uma série de dados foram recolhidos para se poder executar testes estatísticos perante os mesmos e através dos seus resultados, rejeitar ou aceitar as hipóteses previamente expostas e consequentemente avaliar as condições da plataforma após implementação. A avaliação resultou numa apreciação positiva perante todos os indicadores, levando à conclusão de que a plataforma ia de encontro com as expectativas que lhe foram designadas anteriormente. Por fim, foi elaborada uma conclusão que refletia sobre o trabalho desenvolvido (considerando que todos os objetivos da plataforma foram atingidos), o trabalho futuro (breves melhorias e sugestões) e uma apreciação pessoal sobre o projeto.

Acknowledgement

I thank my parents for their constant love and support.

I would like to thank my supervisor Dr. Ângelo Martins, as well as Dr. Nuno Bettencourt and Dr. Ana Maria Madureira for their cooperation, guidance and support.

I would also like to thank my classmates and teammates that throughout the degree collaborated with me.

Finally, I'd also like to thank ISEP.

Contents

List of Figures	xv
List of Tables	xvii
List of Source Code	xix
List of Acronyms	xxi
1 Introduction	1
1.1 Context	1
1.2 Problem	1
1.3 Objectives	2
1.4 Research Methodology	3
1.5 Brief State of the Art	3
1.6 Document Structure	5
2 State of the Art	7
2.1 Context	7
2.2 Literature Review Methodology	8
2.3 Clinical Trials and Regulatory Norms	9
2.4 Overview of Healthcare Data Structure Standards	11
2.5 HL7	12
2.5.1 HL7 Version 2 (V2)	12
2.5.2 HL7 Version 3 (V3)	13
2.5.3 Clinical Document Architecture (CDA)	14
2.5.4 FHIR	15
2.6 LOINC	18
2.7 OGC SensorThings	19
2.8 IEEE 11073	23
2.9 Summary	24
3 Value Analysis	27
3.1 Orientation/Innovation Process	27
3.1.1 Opportunity Identification	28
3.1.2 Opportunity Analysis	30
3.1.3 Idea Generation	31
3.1.4 Idea Selection	32
3.1.5 Influencing Factors	32
3.2 Value	33
3.3 Functional Analysis	34
3.4 Standard Selection	36

4	Analysis and Design	37
4.1	Analysis	37
4.1.1	Stakeholders and Actors	37
4.1.2	Functional Requirements and Use Cases	38
	UC1 - Create Clinical Trial	39
	UC2 - Register Patient	39
	UC3 - Associate Patient to Clinical Trial	39
	UC4 - Incorporate Biosensor Data	40
	UC5 - Persist Bundle of Clinical Data	40
	UC6 - Find Patients By Clinical Trial	40
	UC7 - Find Biosensor Data History By Patient	40
	UC8 - Incorporate Sleep+ and Bedroom+ Data	40
	UC9 - Find Sleep+ and Bedroom+ Data By Patient	40
4.1.3	Non-Functional Requirements	41
4.1.4	Domain	41
4.2	Design	45
4.2.1	The 4+1 View Model	45
4.2.2	Platform Architecture	46
	Logical View	46
	Process View	46
	Development View	47
	Physical View	52
5	Implementation	55
5.1	Technical Decisions	55
5.2	Setup of HAPI FHIR Server	56
5.3	Setup of Management API	59
5.3.1	Models	59
5.3.2	Service	64
5.3.3	Controller	73
5.3.4	Swagger Setup	74
5.4	Running the Platform	76
5.4.1	Running the Server	76
5.4.2	Running the Management API	77
6	Experimentation and Evaluation	79
6.1	Purpose	79
6.2	Indicators	79
6.3	Hypothesis	80
6.4	Evaluation Methodology	81
6.4.1	Process	81
6.4.2	Tools	81
6.4.3	Interpretation of Results	82
6.5	Experiments and Result Analysis	82
6.5.1	System Tests and Acceptance Tests	82
6.5.2	Performance Tests for Consecutive Requests	85
6.5.3	Performance Tests for Bundles	87
6.6	Final Evaluation	90

7 Conclusion	93
7.1 Summary of Fulfilled Work	93
7.2 Review of Fulfilled Work	93
7.3 Future Work	94
7.4 Personal Appreciation	94
Bibliography	97
A FHIR Resource Categories and Full Index	101
B Timed values of Performance Tests for Consecutive Requests	103

List of Figures

2.1	HAPI FHIR Example	16
2.2	The OGC SensorThings Sensing Data Model	20
2.3	The OGC SensorThings Tasking Data Model	21
2.4	Example of Framework	23
3.1	Innovation Process	27
3.2	NCD visualization	28
3.3	Worldwide Medtech Sales Top 15 Categories Total Market (2017-2024)	30
3.4	Worldwide Medtech RD Spend (2011-2024)	31
3.5	Value Proposition Canvas	34
3.6	FAST Diagram	35
4.1	Components of the SHA4ALL PPS1 Platform	38
4.2	Use Case Diagram	39
4.3	Biosensor Data attributes	42
4.4	Management API Domain Model	44
4.5	FHIR-based Domain Model	45
4.6	SSD for Use Case of requirement FR4	46
4.7	SSD for Use Case of requirement FR5	47
4.8	Development View - General testing scenario	48
4.9	Development View - Test of a Service	48
4.10	Development View - Test of a Device	49
4.11	Development View - Test of a Device with User Interaction	49
4.12	Development View - Test of an Application	50
4.13	Development View - Test of a system App + Service(s)	50
4.14	Development View - Test of a system Service(s) + Device(s)	51
4.15	Development View - Test of a monolithic system (App + Service(s) + Device(s)) with user interaction with devices	51
4.16	Development View - Test of a monolithic system without user interaction with the devices	52
4.17	Development View - Test of monolithic system with devices connected to App	52
4.18	Physical View - Deployment diagram	53
5.1	HAPI FHIR JPA Server Architecture (<i>FHIR Server Types 2023</i>)	56
5.2	Management API Swagger Page	75
5.3	Management API Swagger Page Endpoint	76
6.1	Example of Postman System Test	82
6.2	Example of Manual Verification for the Acceptance Test	84
6.3	Boxplot Graph for data obtained from Performance Tests for Consecutive Requests	86

6.4	Boxplot Graph for data obtained from Performance Tests for Bundles . . .	89
A.1	FHIR Resource Categories and Full Index	101

List of Tables

2.1	Example of LOINC codes and the LOINC database structure (Drenkhahn and Ingenerf 2020)	18
2.2	Summary of the Researched Standards	25
3.1	Value For Customer - Benefits and Sacrifices	33
4.1	Functional Requirements	38
4.2	Non-Functional Requirements	41
4.3	Biosensor Data Concept and FHIR Observation Comparison	42
4.4	Sleep+ and Bedroom+ Data and FHIR Observation Comparison	43
6.1	Hypothesis for Experimentation and Evaluation	80
6.2	System Tests Summary Table	83
6.3	Summary of the data of the Performance Tests for Consecutive Requests	85
6.4	Summary of the data of the Performance Tests for Bundles	88
B.1	Timed values of Performance Tests for Consecutive Requests	103

List of Source Code

2.1	HL7 V2 Message Format Example (<i>Caristix HL7-Definition V2 reference site 2022</i>)	13
2.2	HL7 V3 XML Format Example (Spronk 2007)	14
2.3	FHIR Patient JSON Example (<i>FHIR Overview - Developers 2022</i>)	17
2.4	OGC SensorThings Observation JSON Example (<i>OGC SensorThings API Part 1: Sensing Version 1.1 2022</i>)	21
2.5	OGC SensorThings TaskingCapability JSON Example (<i>OGC SensorThings API Part 2 – Tasking Core 2022</i>)	22
2.6	OGC SensorThings Task JSON Example (<i>OGC SensorThings API Part 2 – Tasking Core 2022</i>)	22
5.1	Resource Provider Example	57
5.2	RestfulServlet Example	59
5.3	BiosensorData Model Class	60
5.4	SleepBedroomData Model Class	61
5.5	SleepData Class	62
5.6	BedroomData Class	62
5.7	Temperature Class	63
5.8	Humidity Class	63
5.9	HeadPosition Class	64
5.10	Data Service Logger	64
5.11	Data Service Clinical Trial Creation	65
5.12	Data Service Patient Creation	66
5.13	JSON Body for FHIR Observation	67
5.14	Data Service BiosensorData Persistence	68
5.15	Data Service Sleep+ and Bedroom+ Data Persistence	69
5.16	Data Service Getting Patients by Clinical Trial	70
5.17	Example of the body required by the Bundle method in the Data Service	71
5.18	Data Service Approach to Posting Bundles	72
5.19	Data Controller Extract	73
5.20	Swagger Configuration Class	75
5.21	Commands to locally run the HAPI FHIR Server	76
6.1	Wilcoxon Test performed in R	87
6.2	Wilcoxon Test performed in R	90

List of Acronyms

AAL	Ambient Assisted Living.
API	Application Programming Interface.
b-on	<i>Biblioteca do Conhecimento Online.</i>
CDA	Clinical Document Architecture.
CRUD	CREATE, READ, UPDATE and DELETE.
CTIS	Clinical Trials Information System.
DIM	Domain Information Model.
EHRs	Electronic health records.
FAIR	Findability, Accessibility, Interoperability, and Reusability.
FAST	Functional Analysis and System Technique.
FEI	Front End of Innovation.
FFE	Fuzzy Front End.
FHIR	Fast Healthcare Interoperability Resources.
FURPS	Functionality, Usability, Reliability, Performance and Supportability.
GDPR	General Data Protection Regulation.
HAPI	HL7 application programming interface.
HL7	Health Level Seven.
HTTP	Hypertext Transfer Protocol.
IEEE	Institute of Electrical and Electronics Engineers.
IoT	Internet of Things.
ISEP	<i>Instituto Superior de Engenharia do Porto.</i>
ISO	International Organization for Standardization.
JSON	JavaScript Object Notation.
LOINC	Logical Observation Identifiers Names and Codes.
MEI	<i>Mestrado em Engenharia Informática.</i>

NCD	New Concept Model.
NFC	Near-field Communication.
OGC	Open Geospatial Consortium.
PPS	<i>Produtos, Processos, Serviços e Tecnologias.</i>
PPS1	<i>Plataforma de Desenvolvimento e Ensaios SMART-HEALTH-4-ALL.</i>
REST	Representational State Transfer.
RPC	Remote Procedure Call.
SSD	System Sequence Diagram.
TMDEI	<i>Tese/Dissertação/Estágio.</i>
UI	User Interface.
URL	Uniform Resource Locator.
XML	Extensible Markup Language.

Chapter 1

Introduction

In this introductory section, the context of the document and the software project it entails will be detailed, along with a description of the problem that led to its development and the objectives set to address said problem. There will also be a rundown of the research strategy used, a summary of the state of the art findings and an account of the remaining sections of the report and their contents.

1.1 Context

This document was developed during the course of *Tese/Dissertação/Estágio* (TMDEI), the final course unit of the *Mestrado em Engenharia Informática* (MEI) degree from *Instituto Superior de Engenharia do Porto* (ISEP). The intent of this unit is the development of a software project and its proper documentation in the form of a thesis, utilizing proper professional and technical-practical practices, engineering principles and tools, while respecting a series of modules taught in the unit and applying them in various stages of the project development.

The system that is idealized in this thesis is associated to *SMART-HEALTH-4-ALL* (2023), a project that aims to dynamize the ecosystem dedicated to research, development, production, commercialization and spread of Smart Health medical technologies in Portugal. It is a cooperative effort between 11 companies and 13 Research and Development organizations, of which ISEP takes part.

This initiative is divided in 6 sub-project units, named *Produtos, Processos, Serviços e Tecnologias* (PPS). The contributions of ISEP fit into *Plataforma de Desenvolvimento e Ensaios SMART-HEALTH-4-ALL* (PPS1), which aims to develop a platform for the persistence and manipulation of medical data. This platform is intended to be used in integration tests and clinical trials, with the intent of facilitating the capture, processing, validation and interpretation of the measurements and results, all while respecting the norms and standards expected in the treatment of data by medical software in trial environments.

1.2 Problem

The structures used to define medical and clinical data can be quite varied and dependant on the hardware used and the circumstances surrounding their capture. In the context of large scaled tests and trials of Ambient Assisted Living (AAL) or e-health devices and services, the persistence of health information has requirements and frameworks of specific legal characteristics.

If the integrity of the information passed between devices or individuals is compromised, the consequences can be quite problematic and lead to waste of resources.

As such, a platform that makes use of Healthcare Data Structure Standards to establish a common semantic domain for the input of clinical data from devices, applications and services can safeguard the integrity of the data and secure its adequate structuring.

In a Clinical Trial Management System, such as the one required by the SMART-HEALTH-4-ALL project, the problem of unstable and unverified data transfer between systems is specially relevant and a platform as the one described can bring an abundance of advantages and facilitate various processes in the development of medical devices. It is predicted that it could serve as a beneficial asset that could heighten the speed at which clinical trials are executed and approved and reduce costs consequently.

1.3 Objectives

Knowing the circumstances of the problem, the project should aim to develop a system that can provide tools for users to be able to:

- connect clinical systems (devices, services and applications) and have their measurements imported, revised and validated;
- import multiple measurements and/or relevant clinical data in a single operation;
- develop simpler and more accessible input options that facilitate the insertion of key clinical concepts (such as patients, trials and others) and work with the applied data standards;
- review the database values and their proper structuring according to the applied standards;
- obtain specific query results and raw statistic data based on the input information;
- generate and access consistent automated logs of the operations executed in the platform and their relevant characteristics (user, type of operation, date, information input/output)

In order to achieve the described platform and its functions, a series of goals were established in accordance to the requirements of the unit and of the project itself:

- Analyze the state of the art of the persistence of clinical data, with emphasis on the context of AAL/e-health and Internet of Things (IoT);
- Design, implement and evaluate an infrastructure for the persistence and review of medical information to a testing/trial platform of AAL/e-health devices and services, with the goal of integration of said platform in processes of capture, processing and analysis of said information;
- Utilize open standards for data structuring, specifically the use of Fast Healthcare Interoperability Resources (FHIR).

The previously listed objectives also entails the use of proper research and documentation techniques.

1.4 Research Methodology

The methodology used for the research presented throughout the document was a Systematic Literature Review.

As stated by Tungpunkom (2015), a systematic review is a scientific study of all available evidence on a certain topic. It can be of:

- qualitative nature, where the results of relevant studies are presented and summarized but not mathematically or statistically combined;
- quantitative nature, where statistical methods are used to aggregate the results of multiple studies;
- meta-analysis nature, which uses statistical methods to "[...] integrate estimates of effect from relevant studies that are independent but similar and summarize them".

The research that was executed for this document is of a qualitative nature.

This methodology is composed by the Systematic Review Process, which is to be used in order to find the best possible results, with the most relevant information for the research topic at hand. It is defined by a series of steps:

- Development of a research question, which will be used in the bibliographic databases to obtain relevant studies;
- Use of inclusion and exclusion criteria, which are terms, keywords, rules or conditions that will be applied in relation to the research question (for example, accepting only studies published after 2018);
- Obtaining resulting list of studies from bibliographic databases;
- Selecting studies from the result list, based on certain sections of these, such as their abstracts;
- Determining qualities of the selected studies and removing those that do not meet expectations;
- Extraction of data from the valid studies;
- Analysis of the extracted data and presentation of results;
- Interpretation of obtained results;

These steps are repeated for any topics that must be researched. In this document, this approach was used to obtain information in topics such as data interoperability and health data standards.

1.5 Brief State of the Art

To understand the purpose of the system proposed in this thesis, it is important to denote the objectives of SMART-HEALTH-4-ALL, that include the expediting of the development and release of smart health solutions in Portugal. PPS1, the sub-project which the proposed system is part of, is focused on clinical trials of these solutions, specifically medical devices and services. The devised platform will deal with the persistence and validation of data from these solutions and provide a source of information for other platforms to perform data

analytic reviews. The persistence platform of this thesis should take into account the structuring, storage and provision of clinical data according to data structure standards as well as legal and ethical regulations, while also guaranteeing levels of usability and performance needed for its proper functioning.

By executing a literature review based around conference papers, journal articles, official documentation and information provided by higher-ups in the SMART-HEALTH-4-ALL project, the state of the art starts by looking at the concept of clinical trial, which involves the capture and study of changes in the condition of a group of subjects exposed to an intervention, which can be a medication, treatment, device, among others. A clinical trial entails a great deal of detail and regulatory oversight, with the adherence to legal and ethical standards being of importance, in some cases being actually enforced. For the PPS1 sub-project, and the platform of the thesis, these norms were explored, looking at standards that infer the expected quality of medical software, but also legislation documents that oversee the areas of clinical research, data protection and medical device management. Knowing this, it is highlighted the relevance of technology that compliments the fidelity to these norms, during the development of the platform.

Subsequently, the state of the art looks to the previously mentioned data structure standards, to define these as basis of representation of health data in comprehensive, widely accepted forms, and correlates their importance with the need for interoperability in the communication between medical devices, services and systems. This stems from the various data structures that medical devices and services use to describe the same concepts and ideas, and how systems should aim to guarantee interoperability. In other words, having the systems that connect to these devices and services utilize a specific standard that will ease the input and output of information.

Some of the available and more popular health data standards were then researched. HL7 presented four standards that were of relevance. HL7 Version 2 proved to be a widely implemented standard, being a source of various benefits for the development of medical technology upon its release, but loosing favoritism in recent use, due to its older and more rudimentary approach. HL7 Version 3 showcased dynamic implementations based around XML, providing more robust data persistence through the use of referential data, but faltering in its level of compatibility. HL7 CDA offers the same advantages and disadvantages as Version 3, with added document support. HL7 FHIR, which is the standard that was chosen by SMART-HEALTH-4-ALL, presents a massive improvement over the previous HL7 standards, providing not only the compatibility missing in Version 3 and CDA, but also containing a much richer and complex library of concepts and semantics, support for JSON and XML and the inclusion of a practical, easy to use interface.

Other standards outside HL7 that were explored include LOINC, which depicts a vast compendium of terms, semantics and concepts of the clinical field, widely used and very accessible, but prone to issues like lack of consensus, repeated data and difficulty in maintenance; OGC SensorThings, versatile and useful, although complex and layered, it is divided into two parts, one responsible for the management of clinical data and another for its practical use and testing; IEEE 11073, ideal for personal health systems and institutional medical systems, renowned by its capacity for data synchronization and integrity, but lacking in security.

1.6 Document Structure

Besides the Introduction present in this chapter, the main body of the thesis is composed of six additional chapters.

Chapter 2 exposes the state of the art, which firstly contextualizes the project, its placement in the clinical tech development environment, and secondly reviews the topics of interoperability and health data structure standards, with an in-depth look at some standards and their features.

Then, Chapter 3 focuses on the value analysis, which delves into the innovation process, the value proposed by the project (through the showcase of the Value Proposition) and the Functional Analysis of the capabilities of the proposed platform (through a Functional Analysis and System Technique (FAST) diagram).

Afterwards, Chapter 4 presents the analysis and design stages of the development of the app, according to the principles of Requirements Engineering.

Chapter 5 will look to the implementation of the designed solution and the approaches taken in the use of the data standards that were studied.

Chapter 6 will feature the Experimentation and Evaluation, which comprises an appreciation of the implemented solution in regard to various quality metrics and results from testing the performance of the platform.

Finally, Chapter 7 presents a conclusion to the thesis, with an overview of obtained results and possible improvements.

Chapter 2

State of the Art

For the state of the art, the thematic context of the project will be addressed, followed by a presentation of how the sources of information were obtained, detailing research inputs used and respective results. Once this is done, the relevant topics will be addressed, analysing the regulatory norms applied to clinical trials, complemented by the exploration of healthcare data standards, their importance in relation to interoperability and the characteristics of a selection of these standards, considered for the project implementation.

2.1 Context

The initiative which the platform detailed in this document is part of, *SMART-HEALTH-4-ALL* (2023), is a project veered to the improvement of clinical technology development in Portugal. Its goal is the creation of tools and systems that will promote and expedite the innovations in the field of research and development of Portuguese Smart Health medical technologies and their internationalization.

One of the sub-projects of SMART-HEALTH-4-ALL is the PPS1. It is based around clinical trials, which involve the capture of measured data from various medical devices and services and the validation of said data, meant to be used in the testing of these devices and services to assure their conditions of production and distribution. As such, the intent of the sub-project is to develop software that will speed up this stage of development, reducing the duration and costs of the trials.

The first stage of PPS1 entails the creation of a platform to be used in the execution of the clinical trials. This platform is the one with its development documented in this thesis. Its capabilities must include the input of measured data from various medical devices and services. The data needs to comply to norms and standards that are in coherence with those used in the wider clinical trial environment, and as such, the platform must provide the information and restrictions for the insertion of data, assuring the security, integrity and organization of the information it receives (FHIR was the proposed standard to be used for this, although alternatives will be researched and presented to better understand why it is the favored option in this technological sector).

The data will eventually be exported to other platforms of PPS1 in order to be processed and obtain analytic data. As such, the platform must also be one that facilitates not only the input of data from various sources but its export as well, assuring it is done in a format corresponding to those expected by apps and systems that will execute complex analytic reviews. This means that the platform should provide a series of queries, of different levels of complexity, to facilitate the use of the stored information. It should also have a

consideration of usability, as the intended users will be those in the medical field, who expect ease of access and comprehensive displays of the clinical data, and a consideration for the various norms and legal restrictions that the clinical field of software development requires when dealing with sensible personal data.

The platform should also keep logs of its executions, including dates and details of actions taken, so as to assure a steady documentation of the data manipulation, which is important for the execution of the clinical trials and their own documentation.

To tackle the development of this platform, an understanding of the concepts of clinical trials, the legal and ethical restraints that these involve, as well as standards used to structure e-health data is crucial. By exploring these topics, we can contextualize certain requirements of the platform, the standards that are to be used (FHIR) and the evolution of the sector, while also examining the importance of standards when it comes to the transfer of clinical data between devices and systems.

2.2 Literature Review Methodology

The information gathered for this state of the art has three primary sources:

- Documents provided by the higher-ups in the project, specifically used when exploring regulatory norms in chapter 2.3;
- Official documentation and data provided by the tools and technology explored throughout the chapter;
- Documentation, mostly journal articles and conference papers, found through the use of a systematic literature review.

The documents from the literature review were selected by way of the electronic databases of Google Scholar, IEEE Xplore and *Biblioteca do Conhecimento Online* (b-on).

Initially, the terms applied to the search prompt were "*health data standards*" and "*health-care interoperability*", but these resulted in a collective of 148,000 results. As such, additional conditions were used, such as only requesting papers published in the last 10 years (2013 to 2023) and using keywords such as *IoT*, *internet-of-things*, *legal*, *norms*, *usability*, *security*, *integrity* and *clinical trials*. This helped reduce the result pool to a more manageable number, with keywords being added or removed, depending on the scope of the search that was desired.

This process was repeated multiple times, with modifications according to the specificity of the topic that was being researched. For example, when looking for information regarding standards and health interoperability in current medicine, the conditions described previously were used, but when the research focused on a specific technology or tool, the parameters were modified so that results focused on it.

Each time a result was deemed relevant, the reference was saved, leading to a later review and extraction of the necessary data from it. In most cases, the selected results were those that contained large amounts of usable information, as opposed to only a handful of passages, maximizing the relevance of the referenced material in the thesis.

All in all, there were 18 essential documents that resulted from the review that were used across the following sections of the state of the art, alongside the previously provided data

sources and official documentation of explored technologies and tools (39 references in total).

2.3 Clinical Trials and Regulatory Norms

As defined by Friedman et al. (2015), clinical trials can be seen as "[...] a prospective study comparing the effects and value of intervention(s) against a control in human beings". In other words, a trial involves taking a subject, or group of subjects, and exposing them to an alteration, the so-called "intervention", with the prospect of witnessing, registering and analysing its impact on the usual human condition.

The intervention is the key to the execution and definition of a trial, as explained by Friedman et al. (2015):

"A clinical trial must employ one or more intervention techniques. These may be single or combinations of diagnostic, preventive, or herapeutic drugs, biologics, devices, regimens, procedures, or educational approaches. Intervention techniques should be applied to participants in a standard fashion in an effort to change some outcome."

The importance of clinical trials stems from the benefits that they can bring to the advancement of medicine and health conditions of the public, as "well-designed and sufficiently large randomized clinical trials are the best method to establish which interventions are effective and generally safe and thereby improve public health". However, in order for these trials to even occur or be approved, there are legal and ethical requirements that must be considered. Effectively, "for many clinical trials, including those involving new drugs, devices, or biologics, or new indications for existing interventions, there are national and local regulations that must be followed in order to conduct clinical research" (Friedman et al. 2015).

As most products developed under the SMART-HEALTH-4-ALL project fall into the categories of Medical Devices or Software as a Medical Device, it is expected that these products comply with some international standards as well as ensure easy integration with existing systems using interoperability standards.

Some of the International Organization for Standardization (ISO) standards required by the project are:

- **ISO13485:2016** - Specifies requirements for a quality management system where an organization needs to demonstrate its ability to provide medical devices and related services that consistently meet customer and applicable regulatory requirements. It demands a more complete degree of documentation that considers work environments, risk management and conception control, among other regulatory requirements (*ISO 13485:2016* 2023);
- **IEC62304** - Defines the life cycle requirements for medical device software. The set of processes, activities, and tasks described in this standard establishes a common framework for medical device software life cycle processes: risk management, development, maintenance, configuration and software-related problems solving. The IEC 62304 standard also introduces 3 software safety classes, from A (the least crucial) to C (the most crucial), which the processes must consider (*IEC 62304* 2023)

As for legislation requirements devised to comply with international guidelines and best practices, there were those that deal with clinical research, with the development of medical devices and with data protection.

In regard to the legislation requirements with regard to clinical research, the following must be considered:

- **The Clinical Research Act (Law No. 21/2014)** - Establishes specific provisions regarding the conduct of clinical trials, clinical studies with and without intervention and those using medical devices (“Lei n.º 21/2014, de 16 de abril” 2014-04-16);
- **Regulation (EU) 536/2014** - Concerns the way in which clinical drug research is conducted in the European Union. The Regulation harmonises the procedures for the submission, assessment and monitoring of clinical drug trials in the EU through the Clinical Trials Information System (CTIS) that it details (“REGULATION (EU) No 536/2014 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL” 2014-04-16);
- **The Declaration of Helsinki** - non legally-binding (only morally-binding) document containing set of ethical principles regarding human experimentation (“WORLD MEDICAL ASSOCIATION DECLARATION OF HELSINKI Ethical Principles for Medical Research Involving Human Subjects” 2013-06-15);
- **The International Conference on Harmonization’s Good Clinical Practice standards** - guideline intended to protect the rights of human subjects participating in clinical trials and to ensure the scientific validity and credibility of the data collected in human clinical studies, providing for a more economical use of human, animal, and material resources and the elimination of unnecessary delays in the global development and availability of new medicines, and at the same time maintaining safeguards on quality, safety, and efficacy and regulatory obligations to protect public health (Dixon 1999).

The relevant requirements that consider data protection are:

- **Regulation (EU) 2016/679** - details conditions of protection of natural persons with regard to the processing of their personal data and on the free movement of such data (“REGULATION (EU) 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL” 2016-05-04);
- **Law 58/2019** - Related to Regulation (EU) 2016/679. Assures its application on a national judicial level, assuring the protection of singular personal data in its treatment and circulation (“Lei n.º 58/2019, de 8 de agosto” 2019-08-08);
- **Regulation 798/2018** - Document presenting a list of personal data treatments that are subject to the evaluation of impact on data protection criteria (“Regulamento n.º 798/2018, de 30 de novembro” 2018-11-30)

Finally, the legal regulation that must be considered in the matter of medical devices:

- **The medical devices regulation (EU) 2017/745** - The document details measures that aim to improve the quality, safety and reliability of medical devices placed on the European market, while strengthening the transparency of information related to medical devices for consumers and practitioners and enhancing vigilance and market surveillance of devices in use (“REGULATION (EU) 2017/745 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL” 2017-05-05).

Having now a better understanding of the regulatory requirements that comprise the field in which a project like SMART-HEALTH-4-ALL is inserted in, the choice of technologies,

tools and approaches to the setup of the requested platform should complement not only its functional capabilities and complementary characteristics, like usability and performance, but also the compliance with these legal and ethical norms and regulations.

The platform should adopt design and implementation measures that are capable of meeting the capacity for documentation, data protection and clinical trial/device management that the listed documents call for.

2.4 Overview of Healthcare Data Structure Standards

To understand the relevance of Healthcare Data Structure Standards, one first needs to be aware of the need for interoperability of health records within a clinical setting. As discussed by Iroju et al. (2013):

"[...] interoperability is the ability of different information and communications technology systems and software applications to communicate, to exchange data accurately, effectively, and consistently, and to use the information that has been exchanged".

This study (Iroju et al. 2013) classifies interoperability within the medical field as a growing concern, referencing "[...] the rising cost of healthcare, incessant inefficiencies and healthcare quality failures experienced by healthcare providers and patients" as reasons to focus on the development and betterment of communication and information exchange between medical equipment and responsible individuals, facilitating "[...] data sharing and re-use among disparate healthcare applications and devices, reduction of healthcare costs and the improvement in the quality of care". Moreover, Iroju et al. (2013) identify the biggest obstacle to attaining medical interoperability as the lack of standardization, supported by the elevated complexity of the healthcare domain, the incompatibility in definitions and terminologies among professionals and institutions and the resistance to change demonstrated by these.

As stated in research papers regarding metadata management for health data elements (Yang et al. 2022), healthcare information originating from multiple and varied specialties, devices and domains tends to display formats and meanings of considerable diversity, which can lead to barriers in the communication between heterogeneous systems that manage healthcare-related environments, preventing the desired level of interoperability and stability. As an answer to this problem, the standardization and centralization of health data was pursued over the years.

These standards aim to provide a basis for the representation of relevant health data in comprehensive forms, such as that of model classes and their relationships, allowing for a more generalized and universally-accepted organization of these informational elements, within a clinical context.

One study (Schulz et al. 2019) considers how clinical data is often in critical need of standards due to the circumstances that surround its capture and usage, seeing how "Clinical data are shaped according to the specific needs for which they are collected, such as reporting, communicating, and billing". This correlates with the problem tackled by this thesis, which involves multiple clinical devices making their own series of measurements, with different regards to the composition of certain domain entities that they share, such as patient data and analysis results.

This same study (Schulz et al. 2019) also denotes some of the qualities of data management that a standard should provide by referencing the Findability, Accessibility, Interoperability, and Reusability (FAIR) principles, which describe the importance of descriptive templates that allow for the persistence of data with provenance (so that the context of its creation, who persisted and when, is attached to the data itself), with a consideration for proper and relevant vocabulary and ontology, a proper definition of units and the incorporation of formal languages.

Regarding current standards that manage Electronic health records (EHRs), which Gamal, Barakat, and Rezk (2021) describes as "[...] longitudinal health records persevered in an electronic system to provide safe access to full patient data", many developments have been made, with particular emphasis on the various Health Level Seven (HL7) standards (including FHIR), as well as some others of relevance, such as Logical Observation Identifiers Names and Codes (LOINC), Open Geospatial Consortium (OGC) SensorThings and Institute of Electrical and Electronics Engineers (IEEE) 11073.

As declared by Gamal, Barakat, and Rezk (2021), these current standards present a lot of the characteristics that are required by the current state of EHRs management, tackling issues such as extended quantity and complexity of data with the necessary ease and practicality. As such, the development of medical devices and applications has witnessed a major push for the adoption of these standards.

2.5 HL7

As stated in *HL7 Standards (2022)*, the standards developed by HL7 aim to:

"[...] provide a framework [...] for the exchange, integration, sharing, and retrieval of electronic health information. These standards define how information is packaged and communicated from one party to another, setting the language, structure and data types required for seamless integration between systems. HL7 standards support clinical practice and the management, delivery, and evaluation of health services, and are recognized as the most commonly used in the world".

Among the various developed standards by HL7, there are four whose contributions to the state of interoperability have been considerable: Version 2, Version 3, Clinical Document Architecture (CDA) and FHIR. Each has introduced new concepts and codifications of terms, rules and data dynamics which have improved the information management over their predecessors.

2.5.1 HL7 Version 2 (V2)

This version of HL7 was first developed in October 1987 and its latest update was released in 2011 (*HL7 Version 2 Product Suite 2022*). Its main objectives include the reduction of implementation costs and backwards compatibility to older versions of HL7. It is currently used by 95% of healthcare organizations in the United States of America and is present in more than 35 other countries.

According to *Caristix HL7-Definition V2 reference site (2022)*, Version 2 provides the means for the structuring of messages that will be exchanged, focusing on the content of said messages, it "[...] specifies the exchanged data format only and doesn't determine how data will be transported or how it should be secured."

The exchange of information through messages occurs during a Trigger Event, which *Caristix HL7-Definition V2 reference site* (2022) describes as a provision or request of information in a real-life situation (for example, the request of patient information by a system).

The format of the messages created in this version composes a series of *segments*, each one describing details about the identification of the message contents, the type of Trigger Event, the Patient Identification, details about next-of-kin individuals (optional) and Patient Visit info. An example of said format can be seen in Listing 2.1:

```

1 MSH|^~\&|ADT1|GOOD HEALTH HOSPITAL|GHH LAB, INC.|GOOD HEALTH HOSPITAL
  |198808181126|SECURITY|ADT^A01^ADT_A01|MSG00001|T|2.5.1
2 EVN||200708181123||
3 PID|1||PATID1234^5^M11^ADT1^MR^GOOD HEALTH HOSPITAL~123456789^^^USSSA^SS
  ||EVERYMAN^ADAM^A^|||19610615|M||2106-3|2222 HOME STREET^^
  GREENSBORO^NC^27401-1020
4 NK1|1|JONES^BARBARA^K|SPO^Spouse^HL70063|||NK^NEXT OF KIN
5 PV1|1||2000^2012^01|||004777^ATTEND^AARON^A|||SUR|||7|A0|

```

Listing 2.1: HL7 V2 Message Format Example (*Caristix HL7-Definition V2 reference site* 2022)

Each line of the message starts with a code indicating which segment it is detailing. "MSH" refers to message header, which is responsible for detailing the types of information on each of the other segments. "EVN" corresponds to the Event type, "PID" to Patient Identification, "NK1" to Next of Kin and "PV1" to Patient Visit. After this code, relevant information to each segment is presented. For example, in the "NK1" segment, the consequent sections of the segment describe the name of the individual, their relationship to the patient and their classification as the next of kin, respectively.

Despite its vast use, it has started to be replaced by newer HL7 standards, mostly due to its rudimentary structuring, which does not meet the ideal level of usability and ease of implementation expected in modern technological environments (Schulz et al. 2019).

2.5.2 HL7 Version 3 (V3)

As stated in *HL7 Version 3 Product Suite* (2022), the third version of HL7 improves upon its predecessor considerably, providing "[...] a single source that allows implementers of V3 specifications to work with the full set of messages, data types, and terminologies needed to build a complete implementation". Version 3 aims for a widely used and accepted strategy, working with logic based on model driven methodology and producing outputs in the form of Extensible Markup Language (XML), as opposed to V2, which exclusively manages its information through the use of flexible yet rudimentary message documents that, with the current level of technological development in the clinical field, could be seen as outdated in its usability and practicality, not to mention its difficulty of interpretation.

Among its listed benefits, according to *HL7 Version 3 Product Suite* (2022), Version 3 denotes the inclusion of complete clinical context in its exchange of messages between systems, allowing each one to have a common understanding over the terminologies and semantics of the traded information. The way Version 2 manages and structures data proved to be an obstacle for conformance testing, which led HL7 to develop Version 3 as "a standard that is definite and testable, and provide[s] the ability to certify vendors' conformance".

Version 3 also presents a dynamic level of implementation, so that its standards can be applied to any institution that requires them, be it global, regional or local, as well as its data storage strategy, with Version 3 opting for "robust data repositories rather than as word-processing documents", much more in accordance to modern expectations (*HL7 Version 3 Product Suite 2022*).

It is, however, not always advantageous to choose HL7 Version 3. Due to its new functionalities and improved data manipulation, it is not backwards compatible with older versions of HL7, which means that most systems using HL7 Version 2 or older will have to be upgraded in order to function in a predominant HL7 V3 environment, which can be an extensive and expensive labour (Goossen, William, and Langford 2014).

An example of the XML structuring used in HL7 V3 can be seen in Listing 2.2, where there is an extract of an XML file with an *observationEvent* being detailed. This event corresponds to a glucose test, and it includes not only the results of the measurement, but also multiple other pieces of data that make up the context for it, such as other relevant characteristics (date and time, units, authority that assigned the measurement), reference values to compare to the measured one (and the respective interpretations) as well as various complementary codes (for example, the code for the priority level of the measurement). With this associated context, any system or individual can take this information and properly comprehend it and use it, without the need for any more synchronization of terminologies and semantics, making for an effective transfer of data.

```

1 <observationEvent >
2   <id root="2.16.840.1.113883.19.1122.4" extension="1045813"
3     assigningAuthorityName="GHH LAB Filler Orders"/>
4   <code code="1554-5" codeSystemName="LN"
5     codeSystem="2.16.840.1.113883.6.1"
6     displayName="GLUCOSE^POST 12H CFST:MCNC:PT:SER/PLAS:QN"/>
7   <statusCode code="completed"/>
8   <effectiveTime value="200202150730"/>
9   <priorityCode code="R"/>
10  <confidentialityCode code="N"
11    codeSystem="2.16.840.1.113883.5.25"/>
12  <value xsi:type="PQ" value="182" unit="mg/dL"/>
13  <interpretationCode code="H"/>
14  <referenceRange >
15    <interpretationRange >
16      <value xsi:type="IVL_PQ">
17        <low value="70" unit="mg/dL"/>
18        <high value="105" unit="mg/dL"/>
19      </value>
20      <interpretationCode code="N"/>
21    </interpretationRange >
22  </referenceRange >

```

Listing 2.2: HL7 V3 XML Format Example (Spronk 2007)

2.5.3 CDA

The CDA standard was developed in accordance with Version 3 and aimed to perfect the document management involved in procedures of information exchange within an enterprise (*HL7 CDA® Release 2 Product Suite 2022*). It retains the XML implementation of Version 3, with the same logic of inclusive context declaration and model driven organization, but now

integrates it with "incremental semantic interoperability", which corresponds to documents having different levels of complexity and content depending on the user's declared level of compliance.

Additionally, as stated in *HL7 CDA® Release 2 Product Suite (2022)*, CDA facilitates the archival of relevant clinical files that are not in XML format, by inserting them in the body of the CDA document, which is headed by XML segments, so that machines can interpret the overall document and understand the clinical file it carries and its significance, without having to actually interpret or process that same file. This proves advantageous considering the various formats of clinical files that might be of relevance for certain medical cases, which machines might not be prepared to process regularly.

2.5.4 FHIR

The FHIR Resources, originally named Resources for Healthcare, were introduced in 2011 by HL7, with extensions to previous installments of HL7 standards (Version 2 and Version 3). They feature a Representational State Transfer (REST) Application Programming Interface (API) for exchange of data in the form of JavaScript Object Notation (JSON) or XML (Lehne et al. 2019).

FHIR works with components called resources. As described by Ayaz et al. (2021), "the basic building blocks of FHIR are the so-called resources, a generic definition of common health care concepts (eg, patient, observation, practitioner, device, condition) [...] A resource is the smallest discrete concept that can be maintained independently and is the smallest possible unit of a transaction".

As presented by *FHIR Resource Index (2022)*, Resources extend to five categories, each with 4 to 5 subcategories:

- Foundation (sub-categorized by Conformance, Terminology, Security, Documents and Other), that deals with any pieces of data that relate to basic definitions and settings of the inner-workings of the health system in development;
- Base (its subcategories are Individuals, Entities, Workflow and Management), in it are all the resource types that describe essential domain entities, such as Patients, Organizations or Appointments, to name a few;
- Clinical (with subcategories Summary, Diagnostics, Medications, Care Provision and Request & Response), where the resources that relate to medical and clinical concepts, such as measurements, documents or lesser entities. From these, Observations, a resource type from the Diagnostics subcategory, should be highlighted as it is the one corresponding to measurement results from medical devices;
- Financial (subcategories are Support, Billing, Payment and General) deals with any monetary information, such as accounts and payments;
- Specialized (subcategories are Public Health & Research, Definitional Artifacts, Evidence-Based Medicine, Quality Reporting Testing, and Medication Definition) which contains all the concepts that are part of a very specific or non-general medical field or branch.

A visual representation of these categories, subcategories and the full list of resources that fit each one is present in the Figure on Appendix A.

The use of resources is managed by "[...] standardized browser applications that enable the user to access clinical data from any health care system". FHIR provides the basis for these applications, officially named HL7 application programming interface (HAPI), allowing users to customize them as they see fit. This presents one of FHIR's main advantages: its usability and its adaptability. The user-friendly design and its customization options allow clients to shape the HAPI FHIR so that it fits to the business structure of their institution with relative ease (Lehne et al. 2019).

According to Hussain, Langer, and Kohli (2018), another aspect that strengthens FHIR as an ideal choice for the development of current health systems is its use of JSON and XML, in a RESTful environment, as previously mentioned, which provides a more comprehensible, lighter and more efficient form of data transport:

"[...] FHIR lowers barriers to implementation, especially for developers unfamiliar with legacy healthcare application protocols (e.g., HL7 2.x), due to adherence to RESTful design principles, which are ubiquitous in modern software development [...] FHIR APIs are not only easier to implement in server-to-server communications, but also underpin many mobile and client-side browser applications. The lightweight nature of JSON in particular makes it easier for applications with limited processing power and/or memory (such as mobile phones) to perform well vs. heavier protocols".

In Figure 2.1, one implementation of the HAPI FHIR can be seen, where the list of resource types can be accessed on the left side and on the right is the list of the resources of the Patient type and the JSON format of one of them. The interface also offers accessibility options, such as choosing what additional information is to be shown (Summary) and the choice between XML and JSON presentations.

The screenshot shows the HAPI FHIR web interface. On the left, there are navigation options and a list of resource types. The 'Patient' resource type is selected, showing 7 resources. On the right, a table lists Patient resources with columns for Read, Update, ID, and Last Updated. Below the table, the 'Raw Message' section displays the JSON representation of a Patient resource.

Read	Update	Patient/21	2016-02-21
Read	Update	Patient/9	2016-02-20
Read	Update	Patient/25	2016-02-21
Read	Update	Patient/13	2016-02-21

```

{
  "resourceType": "Bundle",
  "id": "e94158e3-15df-45d9-ab11-b2ce3d940fdc",
  "meta": {
    "lastUpdated": "2016-03-05T16:40:03.596+00:00"
  },
  "type": "searchset",
  "total": 1,
  "link": {
    "relation": "self",
    "url": "http://localhost:8080/hapi-fhir-jpaserver/baseDstu2/Patient"
  },
  "entry": [
    {
      "fullUrl": "http://localhost:8080/hapi-fhir-jpaserver/baseDstu2/Patient/1",
      "resource": {
        "resourceType": "Patient",
        "id": "1",
        "meta": {
          "versionId": "1",
          "lastUpdated": "2016-02-17T19:34:22.000+00:00"
        },
        "text": {
          "status": "generated",
          "div": "<div><div class='hapiHeaderText'><b>TESTCREATERESOURCECONDITIONAL </b></div><div class='hapiPropertyTable'><tbody></tbody></div>"
        },
        "name": {
          "family": [
            "testCreateResourceConditional"
          ]
        }
      }
    }
  ]
}

```

Figure 2.1: Example of HAPI FHIR in execution.

To better understand how information is organized by FHIR using JSON, an example of Patient resource in its JSON format is shown in Listing 2.3:

```
1 {
2   "resourceType": "Patient",
3   "id": "23434",
4   "meta": {
5     "versionId": "12",
6     "lastUpdated": "2014-08-18T15:43:30Z"
7   }
8   "text": {
9     "status": "generated",
10    "div": "<!-- Snipped for Brevity -->"
11  },
12  "extension": [
13    {
14      "url": "http://example.org/consent#trials",
15      "valueCode": "renal"
16    }
17  ],
18  "identifier": [
19    {
20      "use": "usual",
21      "label": "MRN",
22      "system": "http://www.goodhealth.org/identifiers/mrn",
23      "value": "123456"
24    }
25  ],
26  "name": [
27    {
28      "family": [
29        "Levin"
30      ],
31      "given": [
32        "Henry"
33      ],
34      "suffix": [
35        "The 7th"
36      ]
37    }
38  ],
39  "gender": {
40    "text": "Male"
41  },
42  "birthDate": "1932-09-24",
43  "active": true
44 }
```

Listing 2.3: FHIR Patient JSON Example (*FHIR Overview - Developers 2022*)

As the previous listing shows, a resource body follows a specific structure, composed of: the Resource Type (*resourceType*, required), the identification (*id*, required except on create operation), the metadata (*meta*, usually present, common context data to all resources and managed by the infrastructure, missing if there is no metadata), the readable text (*text*, textual representation of the resource), extensions (*extension*, defined by the extensibility framework) and then the actual data that comprises the resource, different for each resource type (in this case, the remaining data that characterize a patient are the name(s), gender, birth date and the flag for if they are an active patient or not) (*FHIR Overview - Developers 2022*).

The interest in FHIR has only grown since its initial release, as its advantages and practical implementations are more and more recognised, with many institutions of relevance priming it as a big step-up for the future of e-health. For example, Ayaz et al. (2021) references how:

" [...] six large technology companies, including Microsoft, IBM, Amazon, and Google, pledged to remove barriers for health care interoperability and signed a letter that explicitly mentions FHIR as an emerging standard for the exchange of health data [...] Using FHIR for the exchange of medical data can provide potential benefits in a large number of domains, including mobile health apps, electronic health records (EHRs), precision medicine, wearable devices, big data analytics, and clinical decision support".

As of the writing of this document, the current release for FHIR is 4.3.0, on which basis the present statements regarding FHIR's domain definitions and functionalities were written. Multiple versions that follow this current one, from 4.4.0 to 5.0.0 are public, but unfinished and labeled as "Work in Progress" (*FHIR Publication (Version) History* 2022).

2.6 LOINC

As defined by Bodenreider et al. (2018), LOINC is a clinical terminology for identifying health measurements, observations, and documents, which was first developed in 1994 by the Regenstrief Institute. It came upon as a solution to the problem of over-reliance in codes and referential data in the standards that were used at the time, such as the previously addressed HL7 V2, without proper registration of these codes and the information that they entailed in a public platform, for the purpose of consensus among institutions. As such, "[...] the LOINC Committee embarked on creating a terminology with the appropriate level of granularity for defining the names of observations used in laboratory and clinical information systems".

This standard works as a database for clinical codes and associates to each the necessary data for its interpretation. To this day, the LOINC standard documentation is revised and published every two years, with the most recent version (released in August 2022) being comprised of 99079 different codes, each corresponding to a different concept in the four areas of knowledge that it entails: Laboratory, Clinical, Attachments and Survey (*LOINC Release Notes* 2022).

The LOINC database of codes follows a tabular structure as can be seen in the example present in Table 2.1.

Table 2.1: Example of LOINC codes and the LOINC database structure (Drenkhahn and Ingnerf 2020)

LOINC code	Component	Property	Time	System	Scale	Method
14682-9	Creatinine	SCnc	Pt	Ser/Plas	Qn	
14684-5	Creatinine	SRat	24H	Urine	Qn	
5802-1	Nitrite	PrThr	Pt	Urine	Ord	Test strip

The *Component* section describes the substance or entity being observed while the *Property* section contextualizes that observation by detailing the circumstances of the measurement.

For example, the first two codes in Table 2.1 describe the measurement of Creatinine, but the first one does so in the context of Substance Concentration (SCnc), while the second one does it in the context of Substance Rate (SRat). The *Time*, *System* and *Scale* sections complement this context by identifying the temporal conditions of the measurement (if it was executed in the moment, Pt, or in a period of time, as in 24H), the sample used (such as Serum, Plasma or Urine, among others) and whether the measurement was Quantitative (Qn), Ordinal (Ord), Nominal (Nom) or Narrative (Nar). The final section (*Method*) is optional and "only included if the measurement technique is clinically significant by affecting the test's result or reference range". In the third code in Table 2.1, it is included to clarify the use of a test strip in the procedure (Drenkhahn and Ingenerf 2020).

While this standard has been widely used and welcomed by many institutions in the past and proved to be useful in times where there was a lack of consensus regarding many clinical semantics, many of its glaring issues have come to light with the rise in the level of scrutiny of medical data management and the growth of requirements for a competent data standard. As stated by Drenkhahn and Ingenerf (2020), LOINC shares some of the same principals and approaches to data management as HL7 V2, but it also shares some of their disadvantages, in particular the exclusion of relevant clinical context, such as result values, measurement units and reference ranges, which the article claims to lead to an availability issue. This same source describes other problems found by practically implementing and experimenting with LOINC, such as the absence of billing data, the reduced context information leading to subjective interpretations of data that should be objective and the difficulty that LOINC has keeping up with the constantly evolving terminologies of genetics diagnostics and other medical fields.

Other sources, such as Carter et al. (2020), corroborate these issues that permeate LOINC, claiming that:

"[...] the flat structure of LOINC precludes data aggregation of related granular concepts under a more general structure. LOINC also do not have a computable infrastructure to enable logical inferences and relationships between parts, terms, and concepts. To achieve data aggregation, LOINC must be mapped with absolute precision and accuracy. Using LOINC to search for tests increases the risk of error and has restricted utility".

Carter et al. (2020) also highlights issues with the actual codes and how "[...] codes have increased duplication and overlap. Non-specific codes and overlapping or unspecified specimen types create confusion. Single results might require multiple codes, and many tests have no code".

2.7 OGC SensorThings

The OGC SensorThings API is a service interface standard that utilizes REST, distinguished for two advantageous characteristics: its succinct and practical data model and its service interface "[...] which supports intuitive and flexible query functions" (Huang, Chih-Yuan, and Chen 2019).

The versatility and usefulness of this standard are well documented, with it being in current use in many institutions and systems of note, private and public, geared towards health data management, such as "[...] air quality data management in Europe by European environment

agency, Smart City Sensors in Hamburg, water quality management of hydrological Network in Baden Wurttemberg and many more " (Santhanavanich, T., and Coors. 2020).

The mentioned data model operated by SensorThings is based on another OGC standard, the OGC Observations and Measurements. It is divided into two parts: one is the Sensing part and the other is the Tasking part (Huang, Chih-Yuan, and Chen 2019).

As described in *OGC SensorThings API Part 1: Sensing Version 1.1* (2022), the Sensing part is focused on IoT devices and applications and their CREATE, READ, UPDATE and DELETE (CRUD) operations. The logic of the entities that comprise the data model of this part is that a Sensor or Thing, which will be the IoT device, will have multiple Datastreams that will constitute a series of Observations. According to the documentation, an *Observation* is "[...] modeled as an act that produces a result whose value is an estimate of a property of the observation target [...] classified by its event time [...], FeatureOfInterest, ObservedProperty, and the procedure used (often a Sensor)". The data model in question can be seen in Figure 2.2.

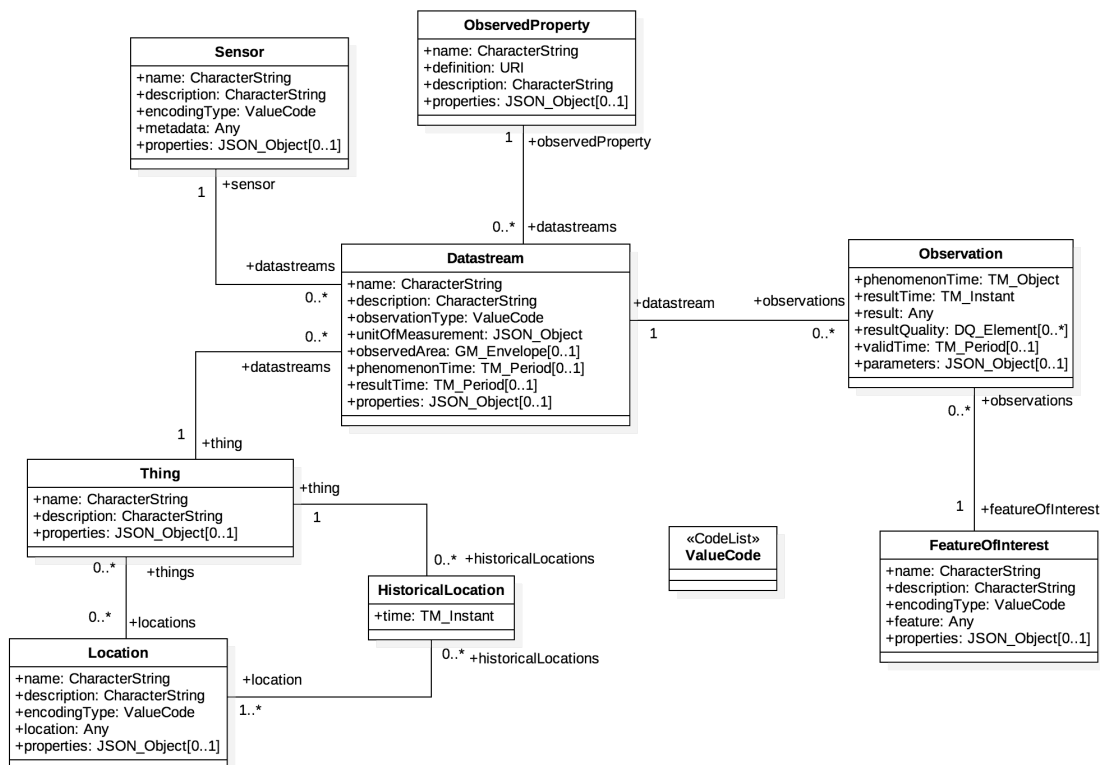


Figure 2.2: The OGC SensorThings Sensing Data Model (*OGC SensorThings API Part 1: Sensing Version 1.1* 2022).

The persistence of the data described in the data model is done through JSON. In Listing 2.4, there is an example of an Observation in the JSON format, where the first four fields identify the Observation itself and other entities that it is related to and its last three fields being its mandatory ones.

```

1 {
2   "@iot.id": 1,
3   "@iot.selfLink": "http://example.org/v1.1/Observations(1)",
4   "FeatureOfInterest@iot.navigationLink": "Observations(1)/
5     FeatureOfInterest",
6   "Datastream@iot.navigationLink": "Observations(1)/Datastream",
7   "phenomenonTime": "2014-12-31T11:59:59.00+08:00",
8   "resultTime": "2014-12-31T11:59:59.00+08:00",
9   "result": 70.4
10 }

```

Listing 2.4: OGC SensorThings Observation JSON Example (*OGC SensorThings API Part 1: Sensing Version 1.1 2022*)

Regarding the Tasking part of the OGC SensorThings API, the documentation *OGC SensorThings API Part 2 – Tasking Core* (2022) describes it as:

"[...] a standard way for parameterizing - also called tasking - of taskable IoT devices, such as individual sensors and actuators, composite consumer / commercial / industrial / smart cities in-situ platforms, mobile and wearable devices, or even unmanned systems platforms such as drones, satellites, connected and autonomous vehicles, etc".

The data model for this part, which can be seen in Figure 2.3, starts by defining Actuators, which are the devices that can be controlled/tasked. Each Actuator will have their functionalities encoded in multiple TaskingCapabilities, with these referencing one or more Tasks, which "[...]" detail the control action that should be run on the task-able device".

The TaskingCapabilities will also be referenced by the Things from the Sensing Part (therefore, connecting the two parts).

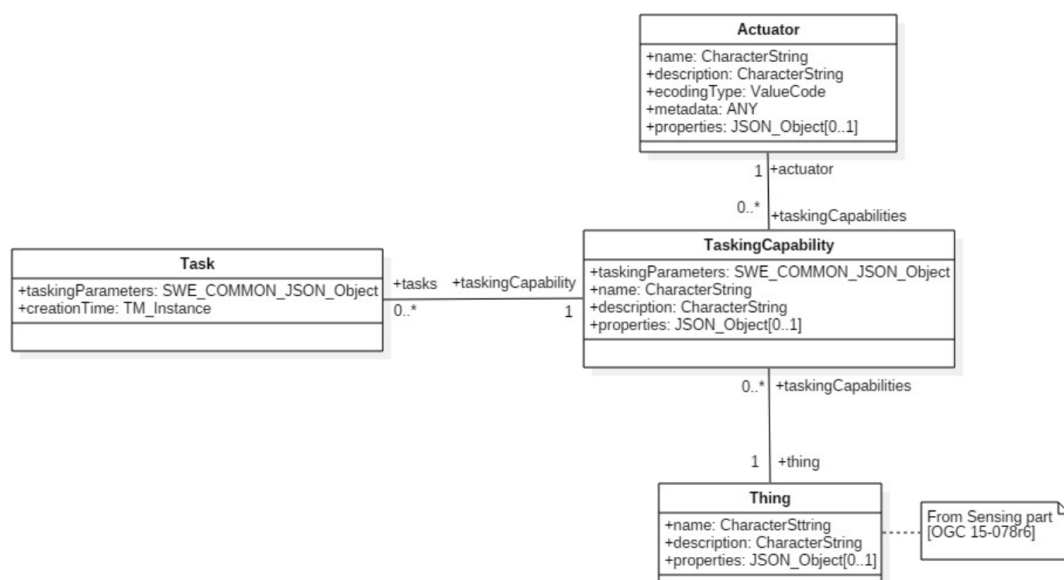


Figure 2.3: The OGC SensorThings Tasking Data Model (*OGC SensorThings API Part 2 – Tasking Core* 2022).

To better understand the concepts of Tasks and TaskingCapabilities, the JSON examples in Listing 2.5 and Listing 2.6 can be of help. The first one describes TaskingCapability, specifically, the capability of a lighting device of turning on and off and of changing its color. The second one presents one of the related Tasks, where one of the colors that the device can use when it is on is declared.

```

1 {
2   '@iot.id': 1,
3   '@iot.selfLink': 'http://example.org/v1.0/TaskingCapabilities(1)',
4   'Thing@iot.navigationLink': 'TaskingCapabilities(1)/Thing',
5   'Actuator@iot.navigationLink': 'TaskingCapabilities(1)/Actuator',
6   'Tasks@iot.navigationLink': 'TaskingCapabilities(1)/Tasks',
7   'name': 'Control Light',
8   'description': 'Turn light on and off, as well as specifying color.',
9   'taskingParameters': {
10    'type': "DataRecord",
11    'field': [
12      {
13        'name': 'status',
14        'label': 'On/Off status',
15        'description': 'Specifies turning the light On or Off',
16        'type': 'Category',
17        'constraint': {
18          'type': "AllowedTokens",
19          'value': ["on", "off"]
20        }
21      },
22      {
23        'name': 'color',
24        'label': 'Light Color',
25        'description': 'Specifies the light color in RGB HEX format.
26        Example: #FF11A0',
27        'type': 'Text',
28        'constraint': {
29          'type': "AllowedTokens",
30          'pattern': "^#[A-Fa-f0-9]{6}|[A-Fa-f0-9]{3}$"
31        }
32      }
33    ]
34  }

```

Listing 2.5: OGC SensorThings TaskingCapability JSON Example (OGC *SensorThings API Part 2 – Tasking Core 2022*)

```

1 {
2   '@iot.id': 2,
3   '@iot.selfLink': 'http://example.org/v1.0/Tasks(1)',
4   'TaskingCapability@iot.navigationLink': 'Tasks(2)/TaskingCapability',
5   'creationTime': '2017-01-01T00:00:00.000Z',
6   'taskingParameters': {
7     'status': 'on',
8     'color': '#FF0000'
9   }
10 }

```

Listing 2.6: OGC SensorThings Task JSON Example (OGC *SensorThings API Part 2 – Tasking Core 2022*)

2.8 IEEE 11073

As a result of the collaboration between ISO and IEEE, came the ISO/IEEE 11073 standards, also known as X73 standards, developed to meet the needs for interoperability in a world with growing dependence in health devices (Laamarti et al. 2020).

As explained by Kasparick et al. (2015), IEEE 11073 specifies a family of standards. This includes IEEE 11073-10101, which defines nomenclature terminologies and clinical definitions, IEEE 11073-10201, which codifies the Domain Information Model (DIM), and many others.

When it comes to the way IEEE 11073 structures its communications, there need to be Agents and Managers, which Badawi, Laamarti, and El Saddik (2019) describe as the devices that provide the data and the devices that receive the data, respectively. One example of this dynamic would be a thermometer, serving as an Agent, sending its information to a mobile phone, which would be the Manager. These communications are not exclusively one-way, as a Manager can send commands to an Agent and request information from it. A visual representation of this can be seen in Figure 2.4, where the various layers of the IEEE 11073/X73 architecture can be seen and the interactions between users, agents and managers are illustrated:

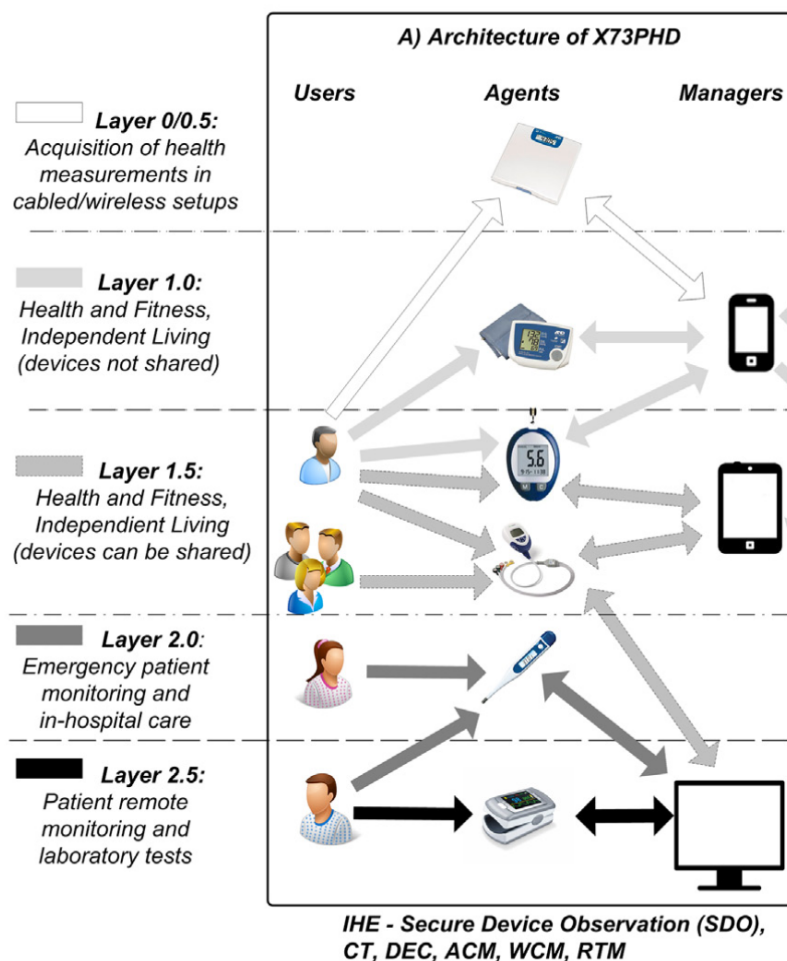


Figure 2.4: Example of Framework (Rubio et al. 2016).

These standards are designed to provide a standardization solution for personal health systems, but have also been used in the design of medical systems for public and private institutions. In fact, many modern researches and projects have delved into this idea. Laamarti et al. (2020) references how these standards have been used for personal implementations, by linking measurement devices, such as blood pressure monitors, weight scales, body temperature sensors and others to mobile devices for personal tracking, while also highlighting the use of these standards by medical institutions in areas like cardiovascular treatment and hypothermic therapy support:

"[...] mobile monitoring of cardiovascular activity and reports its malfunction while preserving the mobility of the patients [...] body temperature sensor for use by patients who require hypothermic therapy [...] as part of the local first aid to monitor a patient's temperatures until the patient reaches the hospital emergency room".

The research developed by Badawi, Laamarti, and El Saddik (2019) corroborates these concepts and ideas, verifying that IEEE 11073 successfully attains the level of interoperability that was sought, both in terms of speed of data synchronization and its integrity, but chides other aspects, like its lack of security measures "[...] or any other related challenges such as the authentication of medical devices and users privacy", which leads to the need for other standards and systems, with Near-field Communication (NFC) technologies being the given example, to be used in conjunction with IEEE 11073 to achieve a stable level and secure implementation.

Lee and Do (2018) also discusses the characteristics of IEEE 11073, pointing out how it may be a useful standard for scenarios involving a personal health device that has relatively small computing power, but not in others that require a bigger scope, complexity and more users and devices involved. In fact, the discussion of this article leads to the conclusion that IEEE 11073 presents less favorable aspects, from its lack of patient information support, to its "Bad Human readability, Hard Learnability, Hard Implementation and Low Extensibility", which contrasts with FHIR, which is seen as having "Good Human readability, Easy Learnability, Easy Implementation and High Extensibility".

2.9 Summary

The SMART-HEALTH-4-ALL project entails the development of systems to support the expansion of Smart Health tech in Portugal. One such system envisions the fast-tracking of clinical trials of the tech, which requires a platform for the storing and structuring of clinical data in such a way that said data is standardized to respect legal considerations and be of easy access for analytical systems to use.

A literature review was executed to determine legal and ethical aspects (regulatory norms) that the storing platform must consider, as well as data structuring standards capable of achieving this, which the platform could use.

The research concluded that Data Standards prove to be of relevance within the world of medical technology, due to the growth of complexity and size of the nomenclatures and semantics involved in it and the need for more uniform and universal representations of crucial clinical data in systems and environments that use them.

For each research standard, their main characteristics and distinguishing factors were presented, as well as the principal benefits and detractors that they would provide the platform with. These are summarized in Table 2.2.

Table 2.2: Summary of the Researched Standards

Standard	Characteristics
HL7 Version 2	Comprised by coded messages; Messages divided into segments, each with a specific attribute; Rudimentary approach to information transfer; Vast use, but actively declining (being replaced with newer HL7 standards);
HL7 Version 3	Logic based on model-driven methodology; Outputs in XML format; Contains complete clinical context in its information trade; Dynamic level of implementation; Robust data repositories;
HL7 CDA	In accordance with HL7 Version 3; Additional document support;
HL7 FHIR	Features REST API; Support for both JSON and XML; Works with components named resources; Provides interface named HAPI; Customization options; Favored by current health systems; Large library of medical semantics;
LOINC	Clinical terminology directory; Each code refers to a series of characteristics and qualifications; Aims for consensus on nomenclature in medical technology; Revised and published every two years; Once very popular, but has declined in use; Lack of clinical context, which leads to too much subjectivity of interpretation; Issues of duplication and overlap with its entries;
OGC SensorThings	Utilizes REST; Succint and practical data model and service interface; Currently popular among health data management organizations; Utilizes JSON ;
IEEE 11073	Defines nomenclature terminologies; Ideal for personal health systems; Thrives in speed of data synchronization and integrity; Lacks authentication measures and user privacy protocols; Criticized for its readability and learnibility;

Chapter 3

Value Analysis

Throughout this section of the thesis, there will be a breakdown of the value of the platform, which questions the relevancy of the idea behind the platform and if its development creates the value expected by the organization developing it. This is done by implementing the process described by Rich and Holweg (2000), which includes the definition of the product (Orientation), the value according to its Value Proposition and its functions and their correlations (Functional Analysis).

3.1 Orientation/Innovation Process

Innovation corresponds to the introduction of new products and ideas in a sector, and is ever present when considering the culture of growing technological development of our time. As such, the focus on assuring these new ideas are of a certain level of quality is of great concern to the business growth. With this, the innovation process, presented in Figure 3.1, describes three steps that should be considered in the development of a innovative product or service.

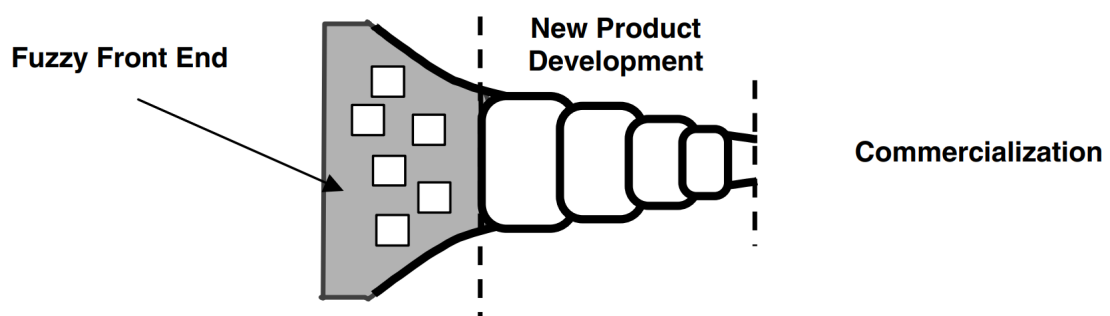


Figure 3.1: Innovation Process (Koen et al. 2001).

As explained by Rich and Holweg (2000), the orientation details the resources required to determine the solutions available that can generate value or opportunities in a specific market. The first step of the process, as presented in Figure 3.1 is the Fuzzy Front End (FFE) technique (or (Front End of Innovation (FEI))). This is then followed by the actual development of the product and its commercialization.

Rich and Holweg (2000) explain how the FFE model can be unstructured and unpredictable to follow and how its usage has decreased as a consequence. Because of this, the use of New Concept Model (NCD) as an alternative is suggested, due to it being more comprehensive

and of easier application, as well as its steps and features adhering adequately to the theme of this thesis.

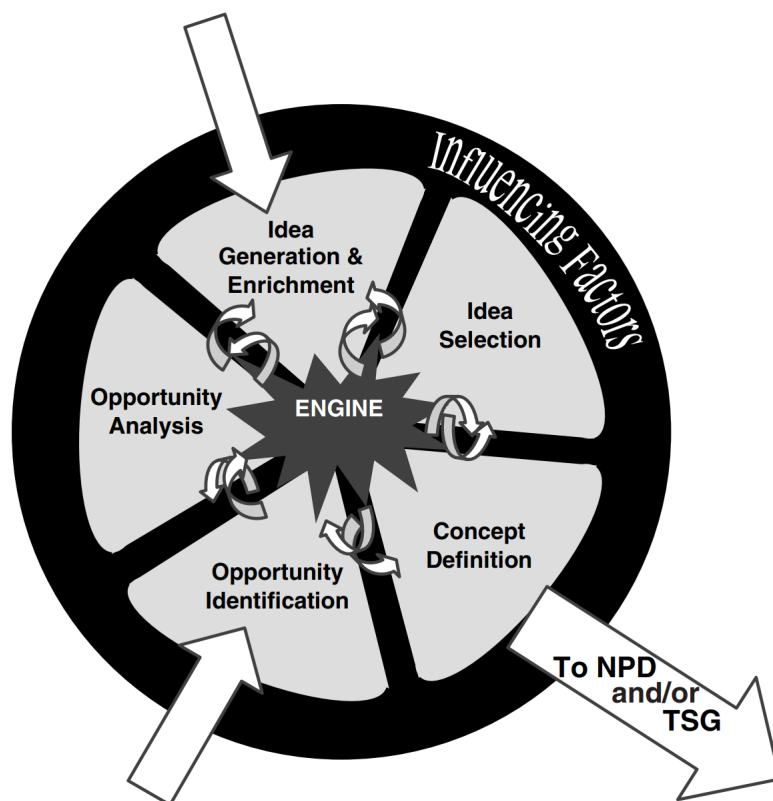


Figure 3.2: NCD Visualization (Koen et al. 2001).

The NCD model, shown in Figure 3.2, describes an organizational organism, where five key activities are driven by an engine (which is understood as the company's business strategy) and are influenced by external Influencing Factors that may cause all sorts of changes in the innovation process.

The five key activities consist of the management of opportunities within a sector, which might lead to the creation and selection of ideas, which in turn may become concepts that will serve as basis for the remaining steps of the Innovation Process (New Product Development and Commercialization).

To apply the NCD model, the the five key activities and the influencing factors will be explored, within the context of the platform in development.

3.1.1 Opportunity Identification

The activity of Opportunity Identification entails finding situations in key sectors where a company's involvement might be of their interest.

As described in Chapter 2.1, *SMART-HEALTH-4-ALL* (2023) aims to make the most out of Portugal's potential for Smart Health Technology development. To achieve this, the project intends to boost technological support and tools and streamline production of devices and software that will impact the medical field. Specifically, the sub-project of *SMART-HEALTH-4-ALL* where the proposed support platform is inserted, PPS1, aims to facilitate

the testing stage of these products to get them approved and in the market with more efficiency.

Knowing this, the identification of the opportunity for this project needs to look at the circumstances in the sector of medical technology that lead the companies and organizations that make up SMART-HEALTH-4-ALL to invest in this venture.

In fact, *Fraunhofer Portugal* (2023) presents two groups of opportunities that boost the purpose of SMART-HEALTH-4-ALL. When it comes to the demand side of the sector, the existing opportunities are listed as:

- growth, increased longevity and aging of the world population;
- changes in epidemiological patterns (linked, for example, to the progressive increase in the incidence/prevalence of oncological, neurological, brain-cardiovascular, metabolic, osteoarticular diseases, etc.);
- more informed citizens/customers with greater digital literacy, more demanding, and greater geographic mobility;
- increased purchasing power in populous countries and with emerging economies;
- need to contain costs ("doing more with less") in developed countries."

In regards to the On the supply side, the opportunities that were presented as driving factors were:

- intense scientific and technological development;
- development of solutions in emerging areas, such as digital health, artificial intelligence, nanomedicine, etc.;
- tendency to change the paradigm of health/disease management, towards a more predictive, preventive, personalized, participatory, and pay-for-performance medicine, under models of healthcare and well-being. increasingly integrated and centered on the individual.

The data persistence platform of this thesis is only a section of this project and its implementation is justified by its own set of opportunities, such as:

- the rising complexity of nomenclatures, semantics and components that make up the knowledge bases of medical devices and systems require transfers of data that use the best and most up-to-date terms, norms and rules of engagement;
- the containment of clinical data sourced from multiple sources (devices and services) in a single server with strong paradigms of security and usability is of high interest to developers and testers of medical technology;
- the increasing need for tools that bridge the capture of measurements and resource input with other systems that can utilize this data for multiple purposes, including analytic reviews;

All in all, *Fraunhofer Portugal* (2023) describes these listed opportunities as crucial to warrant the development of the SMART-HEALTH-4-ALL initiative:

"Many of the trends and opportunities listed converge to justify an increasing commitment to the development and dissemination of technologies and solutions (including

products, processes, systems, and services) that can be generically classified under the concept of Smart Health. This may include a wide spectrum of medical technologies, including medical devices and digital health solutions based on information, communication, and electronic technologies and, in certain segments, on future and emerging technologies, such as nanotechnologies."

3.1.2 Opportunity Analysis

Having determined opportunities, there is the need for a validation of these through the research of data that proves and backs up their claims.

Among the opportunities listed, many reference the increase in interest behind medical tech and its practical application, development and evolution. As stated by EvaluateMedTech (2018), the medical technology market was worth 405 billion US dollars in 2017, with the value for 2024 being forecast to approximately 600 billion US dollars, corresponding to a compound annual growth rate of 5.6%. In fact, the various areas within medical technology that the SMART-HEALTH-4-ALL project aims to exercise in, such as Cardiology, Diabetic Care or Orthopedics, are all predicted to have significant growth, both in sales and in their market share value, as presented in Figure 3.3:

Worldwide Medtech Sales by EvaluateMedTech® Device Area: Top 15 Categories & Total Market (2017 & 2024)

Source: Evaluate, September 2018

Rank	Device Area	WW Sales (\$bn)		CAGR % Growth	WW Market Share			Rank Chg. (+/-)
		2017	2024		2017	2024	Chg. (+/-)	
1.	In Vitro Diagnostics (IVD)	52.6	79.6	+6.1%	13.0%	13.4%	+0.4pp	-
2.	Cardiology	46.9	72.6	+6.4%	11.6%	12.2%	+0.6pp	-
3.	Diagnostic Imaging	39.5	51.0	+3.7%	9.8%	8.6%	-1.2pp	-
4.	Orthopedics	36.5	47.1	+3.7%	9.0%	7.9%	-1.1pp	-
5.	Ophthalmics	27.7	42.2	+6.2%	6.8%	7.1%	+0.3pp	-
6.	General & Plastic Surgery	22.1	34.3	+6.5%	5.5%	5.8%	+0.3pp	-
7.	Endoscopy	18.5	28.3	+6.3%	4.6%	4.8%	+0.2pp	+1
8.	Drug Delivery	18.5	25.3	+4.6%	4.6%	4.3%	-0.3pp	-1
9.	Dental	13.9	21.6	+6.5%	3.4%	3.6%	+0.2pp	-
10.	Diabetic Care	11.7	19.8	+7.8%	2.9%	3.3%	+0.4pp	+3
11.	Wound Management	13.0	17.8	+4.6%	3.2%	3.0%	-0.2pp	-1
12.	Healthcare IT	11.8	17.6	+5.9%	2.9%	3.0%	+0.1pp	-1
13.	Neurology	8.6	15.8	+9.1%	2.1%	2.7%	+0.5pp	+3
14.	Nephrology	11.7	15.6	+4.2%	2.9%	2.6%	-0.3pp	-2
15.	Ear, Nose & Throat (ENT)	8.9	13.1	+5.7%	2.2%	2.2%	+0.0pp	-
	Top 15	342.0	501.7	+5.6%	84.4%	84.4%	-0.0pp	
	Other	63.1	92.9	+5.7%	15.6%	15.6%	+0.0pp	
	Total WW Medtech Sales	405.0	594.5	+5.6%	100.0%	100.0%		

Note: Analysis is based on the top 300 medtech companies. Sales in 2017 based on company reported data. Sales forecasts to 2024 based on a consensus of leading equity analysts' estimates for segmental sales.

Figure 3.3: Worldwide Medtech Sales Top 15 Categories Total Market (2017 2024) (EvaluateMedTech 2018).

This predicts that more and more devices of medical tech are set to be produced and commercialized in the near future, which naturally raises interest in the development support frameworks and tools that integrate the SMART-HEALTH-4-ALL project, including the data persistence platform of this thesis.

The previous opportunities are also supported by the values of investment in research and development of medical tech, which is also predicted to rise by 2024, with 3.9 billion US dollars in expenditure, as shown in Figure 3.4:

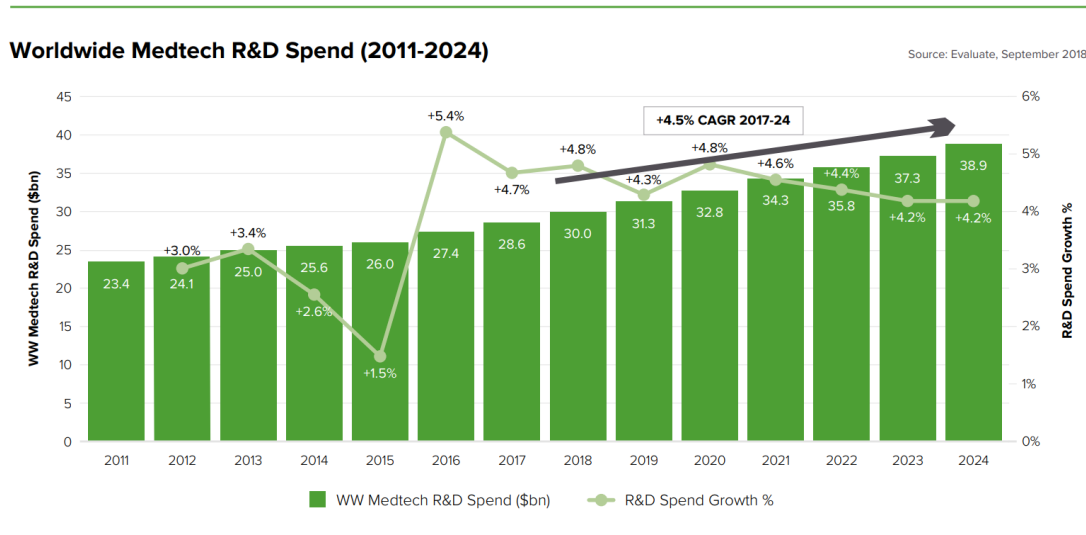


Figure 3.4: Worldwide Medtech R&D Spend (2011-2024) (EvaluateMedTech 2018).

3.1.3 Idea Generation

With an understanding of the opportunities and with the data to back them up, ideas can start flowing and these should be presented here and justified. In the case of this thesis, the ideas will focus exclusively on the platform it encompasses.

Starting with the basic idea of the platform, this should be an API open to receiving information from external sources regarding medical concepts and interpreting said information. This corresponds to the basic concept of the platform acting as a landing dock for the measurements and clinical data provided by devices and services in trial.

The second idea relates to the logging of the received data and intent behind the requested action, registering this way a history of activity within the platform.

A third idea would be the development of a front-end for the API to facilitate the overview of the generated logs and to provide additional information and support in regards to connecting devices and systems to the API and how to send information to it.

The treatment and use of the received data comprise the fourth idea. The app should be capable of constructing JSON objects, which can vary from simple to highly complex, to meet the expectations of the FHIR standards (since FHIR is the suggested standard to use) and consequently send the restructured data to a FHIR Server.

The FHIR server itself is the fifth idea. Since it is the standard that will most likely be used, the implementation of a FHIR server to keep the restructured data and make the most out of the provided FHIR framework and tools.

The sixth and final idea was the development of specific original queries and input options so as to customize the management of information and facilitating the input and output of

relevant information for the users of the platform. One of the examples discussed for input options was the creation of patients within trials, not requiring any additional information besides simple identification of the patient and the trial, leading to a lightweight operation as opposed to the heavier default one. An example of a customized query that could be developed was the listing of patients associated to a specific trial, something not possible using default queries.

3.1.4 Idea Selection

Taking the generated ideas, these should be weighed and considered against a variety of factors that will help classify their viability and their relevance, with the chosen ones becoming concepts to be implemented.

As the decision-making regarding which ideas to select was not up to the developer of the platform but the higher-ups, these were presented and discussed with them and the generated ideas were all selected with the exception of the third idea. It was argued that only the API was required and the front-end would not be beneficial, with a Swagger page for the API being more than capable of exerting the functions that were idealized for the front-end segment of the platform.

As such, the selected ideas lead to the following concepts:

- An API capable of receiving and interpreting HTTP requests with data from medical devices and services;
- The logging of received data, action intents and date of execution;
- The management of FHIR standard structures and generation of corresponding JSON bodies;
- Implementation of a FHIR server;
- Development of customized input options and queries in the API for communication with the FHIR Server.

3.1.5 Influencing Factors

In this step, elements or entities external to the project must be considered if they might have an effect over the project.

The first factor to consider is the versions of FHIR. As described in chapter 2.3.4, FHIR tends to update the versions of their software as well as their knowledge base quite regularly. As such, there should be a constant overview of the current and future states of FHIR and decisions must be made when any new versions are released, in regard to their potential adoption or not.

The second factor to consider are the legal norms and regulations that a system that deals with medical data, specially data related to patients, must consider. By using FHIR, a lot of these precautions are already considered, seeing how FHIR maintains a strict mandate over the required application of such norms and regulations. However, there should still be consideration and oversight over any such legal matter that FHIR does not cover and that may have serious impact on the viability of the opportunities, ideas or concepts that were established.

3.2 Value

The significance of value within a business and its operations is highlighted by Nicola and J. J. Pinto Ferreira (2012):

"The creation of value is key to any business, and any business activity is about exchanging some tangible and/or intangible good or service and having its value accepted and rewarded by customers or clients, either inside the enterprise or collaborative network or outside."

These concepts are further supported by the idea of Perceived Value, which Zeithaml (1988) views as "[...] the consumer's overall assessment of the utility of a product based on perceptions of what is received and what is given".

Knowing that value involves giving as much as forfeiting goods, a value analysis would not be complete without looking at these same segments, when applied to the platform in development, to determine what the actual Value for the Customer is. These can be seen in Table 3.1. It takes into account the advantages brought to those who will be responsible for using the platform in the field, particularly the help it will provide in the overall storage, management and retrieval of clinical data, while acknowledging less favorable aspects, such as the likelihood of significant costs of development and upkeep of the service.

Table 3.1: Value For Customer - Benefits and Sacrifices

Benefits	Sacrifices
Streamlined persistence of data	Cost of maintaining updated FHIR server
Automation of data structuring of measurements and clinical data according to widely adapted standards (FHIR)	Cost of the cloud platform
Adherence to legal and ethical norms and regulations of clinical data management	
Easier retrieval of information through queries	

The value can also be examined by applying the Value Proposition methodology, which is described by A. Osterwalder and Pigneur (2013) as "the definition of how items of value, such as products and services as well as complementary value-added services, are packaged and offered to fulfil customer needs".

The value proposition can take the shape of a statement or can be presented using the Value Proposition Canvas (Osterwalder et al. 2014).

The Value Proposition Canvas is formed around two parts – customer profile and value proposition - each with three sections.

The Customer Profile is composed of:

- Gains – the benefits the customer expects and needs, what would please customers and may increase the likelihood of them adopting a value proposition.

- Pains – the negative situations or risks that the customer is exposed to while getting the job done.
- Customer jobs – the functional tasks customers are trying to perform or problems they are trying to solve.

The value proposition corresponds to:

- Gain creators – how the product or service creates customer gains and how it offers added value to the customer.
- Pain relievers – a description of exactly how the product or service alleviates customer pains.
- Products and services – the products and services which create gain and relieve pain, and which underpin the creation of value for the customer.

Applying this methodology to the platform in question resulted in the canvas in Figure 3.5:

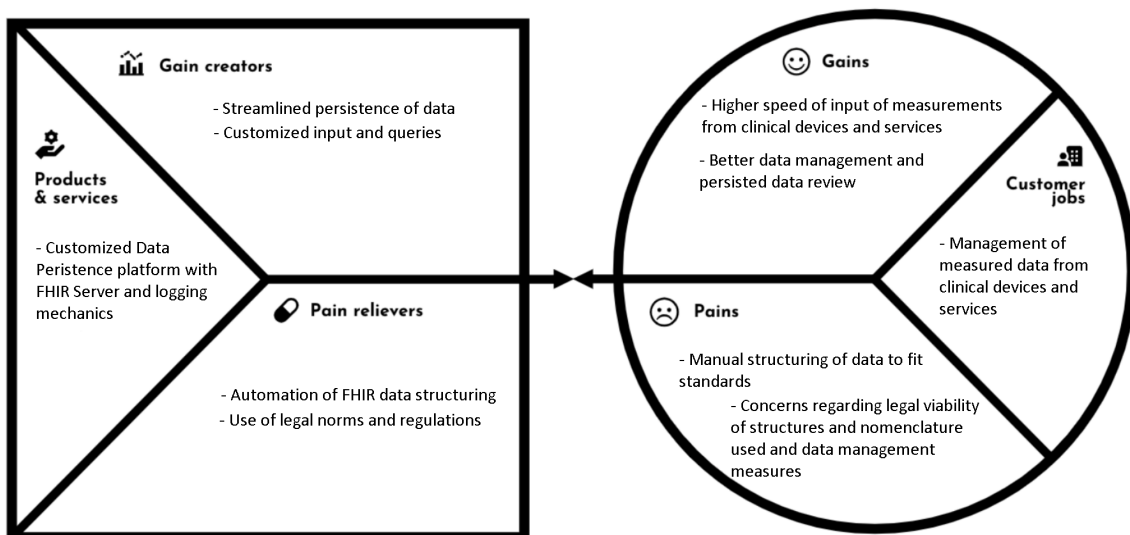


Figure 3.5: Value Proposition Canvas

3.3 Functional Analysis

The functional analysis, as broken down by Rich and Holweg (2000), corresponds to the "[...] analysis of the product by identifying systematically the most important functions of a product or service". This stage of the overall Value Analysis process serves to isolate these functions objectively and establish a hierarchy over their importance and the circumstances of their occurrence, providing support to later stages of their development, such as their implementation and testing.

One way to achieve this analysis is by applying the FAST method, which is "[...] a rigorous method for understanding complex systems by converting the "activities" performed in a system to the "functions" performed by the system for its customers" (Bartolomei and Miller 2001).

The application of FAST leads to the creation of a diagram where functions, presented as simple two-word, verb-noun descriptions of activities of the platform, are displayed in a two-dimensional axis, with their placement and connections between corresponding to their relationships and dependencies (Borza 2011).

The purpose of this is to present as simply and directly as possible what these functions are and how, why and when they come to be:

- "What?" - Corresponds to the activity represented by the function, in a two-word, verb-noun label;
- "How?" - To understand how a function can possibly happen, one can look to the functions at the right of it. The function we are analyzing depends on these and can only happen when they are fulfilled;
- "Why?" - To understand the purpose of a function on the larger scale of things, one can look to the left of it to see other functions that depend on it and consequently justify its existence;
- "When?" - By looking at the vertical placement of functions, the temporal dependencies or ideal order of execution of these functions is made clear. Functions that appear below another should be executed after the one above.

The application of this method to the platform in development resulted in the diagram of Figure 3.6:

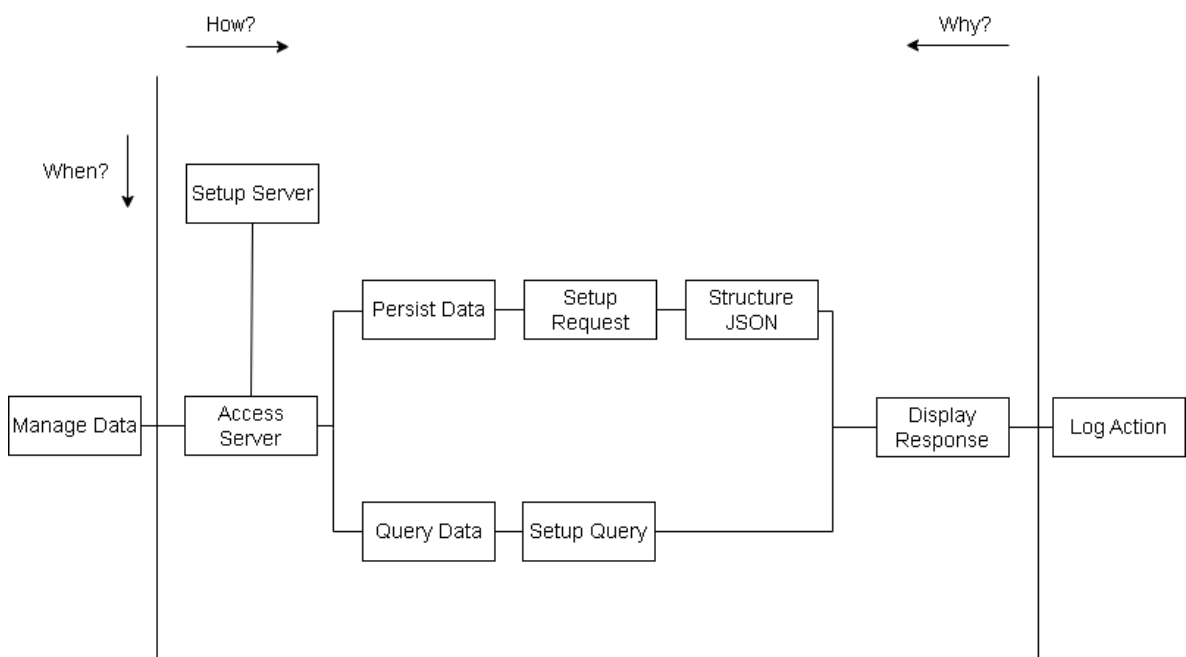


Figure 3.6: Fast Diagram

As such, the list of functionalities for this section, and their significance, is:

- Manage Data: Recognize the intent of modifying or viewing clinical data;
- Setup Server: Check if server is up and take action if not;
- Access Server: Connect to the server to manage its data;

- Persist Data: Insert or modify data in the FHIR server;
- Setup Request: Create request properties for communication with FHIR Server;
- Structure JSON: Create JSON object according to request requirements;
- Query Data: Retrieve data from the FHIR Server;
- Setup Query: Construct query and request according to the information needed;
- Display Response: present console output of the result of communication with the FHIR Server (either result(s) of the input/modification of data or of the query);
- Log Action: Add log corresponding to the action taken, date of action and results to log directory;

3.4 Standard Selection

Having determined the functions and their correlations, the next step in value analysis would be determining which technology to use among the ones studied in the state of the art, for their implementation. Specifically which data standard to use, according to calculated values for each, regarding their importance. However, with the conditions of the project and circumstances surrounding its management, the decision of using FHIR had been already made before the project began. As the choice had been set as part of the project problem, this step was not seen as required for this particular value analysis.

Chapter 4

Analysis and Design

For this chapter, the start of the development of the conceptualized platform will be addressed. For the analysis section, the list of determined use cases will be presented and explained, followed by a look at the domain concepts with which the platform will be working. The design will focus on the organization and architecture of the system, supported by documentation regarding implementation plans and further exploring the use cases that will comprise the platform.

4.1 Analysis

The analysis of the platform revolved around identifying the stakeholders and actor involved in the project, determining the requirements, both functional and non-functional, and establishing the use cases, their details and the relevant domain entities that need to be addressed.

4.1.1 Stakeholders and Actors

For the PPS1 sub-project in which the platform is integrated, the following stakeholders were devised:

- ISEP - as a part of the development team for SMART-HEALTH-4-ALL, specifically, PPS1;
- SMART-HEALTH-4-ALL Collaborators - organizations and institutions associated to or with investment on the overall project;
- Test Managers – people responsible for product testing;
- Clinical Partners – doctors responsible for the clinical component of the test of a product or service;
- Test Designers – people who design and configure the test suites of the products and services;
- Testers – people who runs the test suite;
- Test Analysts – people who analyse test suite results;
- Product Developers – developers of the product or service being tested;
- Users – people using the product or service in a test.

The platform is intended to be used by one single class of actors, the Health Professionals, corresponding to the Clinical Partners stakeholders. These actors will use the platform to inform the tests that are setup, executed and analyzed by other platforms in the PPS1 environment. This organization can be seen in Figure 4.1, where the leftmost platform, the Health Data Integration Platform, corresponds to the one envisioned in this thesis:

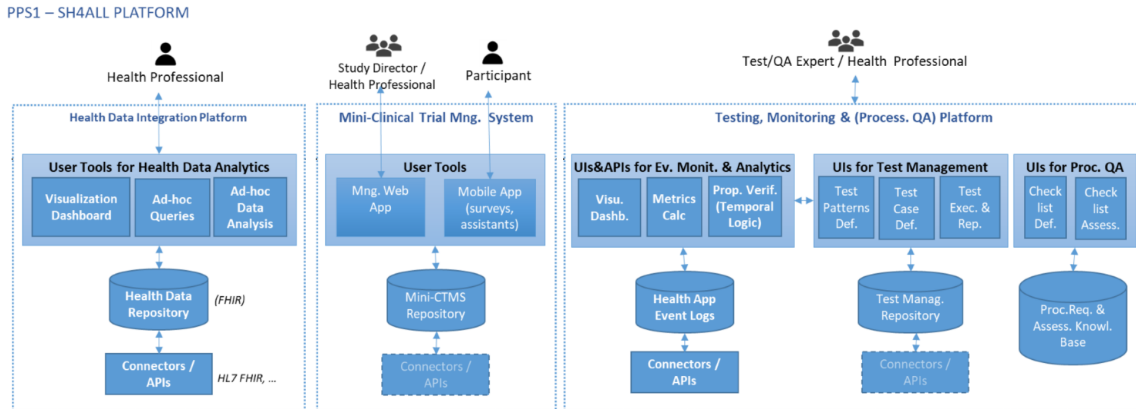


Figure 4.1: Components of the SHA4ALL PPS1 Platform

4.1.2 Functional Requirements and Use Cases

Functional Requirements are used to understand and define a system’s functionalities and behaviors. These are often represented as tasks, services or user stories that the system must implement in order to perform as expected.

For the platform, the requirements were based on some of the functions determined in Chapter 3.3 and on discussions that occurred with some of the stakeholders and other developers. They are presented in Table 4.1:

Table 4.1: Functional Requirements

ID	Actor	Requirement
FR1	Health Professional	Create Clinical Trial
FR2	Health Professional	Register Patient
FR3	Health Professional	Associate Patient to Clinical Trial
FR4	Health Professional	Incorporate Biosensor Data
FR5	Health Professional	Persist Bundle of Clinical Data
FR6	Health Professional	Find Patients By Clinical Trial
FR7	Health Professional	Find Biosensor Data History By Patient
FR8	Health Professional	Incorporate Sleep+ and Bedroom+ Data
FR9	Health Professional	Find Sleep+ and Bedroom+ Data By Patient

The Use Cases that come from these functional requirements can be seen in the Use Case Diagram in Figure 4.2.

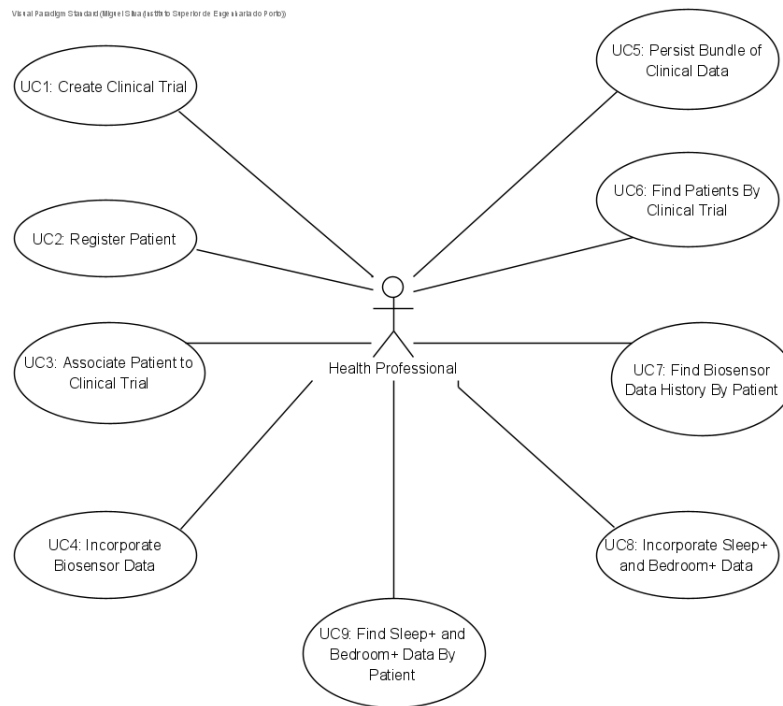


Figure 4.2: Use Case Diagram

UC1 - Create Clinical Trial

As explained in previous sections, the platform intends to store data captured from devices and services that are to be tested in order to fulfill the clinical trial part of their development. This Use Case focuses on the creation of a Clinical Trial from which other relevant information for the testing of a device will cascade (patients and measurements, for example).

As it stands, the envisioned scenario is for the user to request the creation of a new trial, which the system will create and respond with the generated information, specifically the new trial identification, so that the user can attach information to it in later scenarios.

UC2 - Register Patient

A Health Professional will want to register Patients that will participate in trials, providing certain degrees of information to the platform and receiving the generated identification of this newly registered patient.

UC3 - Associate Patient to Clinical Trial

Patients are intended to participate in Clinical Trials, and once both of these are defined, a Health Professional intends to associate a Patient to a Clinical Trial.

Due to a trial involving research of the effects of a device or service on a group of people, a trial can be attached to multiple patients at once.

A Patient, however, can be associated to only one trial at a time. This is because, if a patient was involved in more than one trial simultaneously, the effects of one device in one trial would affect, and consequently invalidate, the results from another trial, related to a different device.

UC4 - Incorporate Biosensor Data

Biosensors will be used to execute measurements of a Patient's vital readings, such as heart rate or respiration rate. These readings will then have to be persisted by a Health Professional, who must indicate not only the dates of their captures, but also additional information, such as the stance of the patient during measurement and the patient identification number so that this information can be associated to a given patient in the server.

UC5 - Persist Bundle of Clinical Data

A *Bundle* (2023) is defined by FHIR as a "[...] list of resources with some context". It can be understood as a single resource that is composed by a series of other resources, which means that a bundle can be a collection of multiple patients, trials, measurements and other FHIR resources in one single packaging. Its purpose is to allow the persistence of multiple entities in one single request, as opposed to a request per entity.

It can be advantageous as it may be faster and lighter on resource use (less HTTP requests being executed), but also has considerable drawbacks. The most notable of these are: (I) the possibility that a bundle persistence might result in an overly heavy request, due to the amount of information it may contain; (II) the setup of a bundle might be too complex for the average Health Professional and became a hindrance to their efficiency.

Nevertheless, it is still relevant to have the use of bundles in the platform as there can be circumstances during the execution of a Clinical Trial (such as time constraints or large amounts of resources to be persisted) where the characteristics of this request type might be favorable.

UC6 - Find Patients By Clinical Trial

Having previously described the association of patients to trials, during sections of the trial where the retrieval and review of information must be done, queries such as the one pertaining to this Use Case, are essential. By providing the system with the identification of a specific trial, the Health Professional must receive a list of all the patients associated to it and their respective data.

UC7 - Find Biosensor Data History By Patient

Similarly to the previous case, this query is also meant to be used in the review of trial data and offers the Health Professional the option to view all the registered Biosensor Data that was obtained for a given patient.

UC8 - Incorporate Sleep+ and Bedroom+ Data

This use case relates to the capture of measurements made by AAL devices and systems, that aim to monitor the sleeping and overall living conditions of a patient, overlooking the temperature and humidity of the room and bed where the patient lies, as well as the position of the patient's head, at a given interval of time.

UC9 - Find Sleep+ and Bedroom+ Data By Patient

Similar to UC6 and UC7, used to gather the measurements made for Sleep+ and Bedroom+, regarding a patient and a specific point in time, so the data can be used in analytic reviews.

4.1.3 Non-Functional Requirements

Non-Functional requirements are specifications that describe the system's capabilities and constraints and attempt to improve its functionality. These are usually not assigned to actors, but instead are considerations that the development team must have when designing and implementing the system. For the platform these are listed in Table 4.2, adapting the Functionality, Usability, Reliability, Performance and Supportability (FURPS) model:

Table 4.2: Non-Functional Requirements

ID	Attribute	Requirement
NFR1	Functionality	Logging: The system shall provide historical information about the management of clinical data
NFR2	Usability	Ease of input of data for users not versed in JSON
NFR3	Usability	Ease of viewing of requested information
NFR4	Reliability	The system must be monitored to allow for quick recovery from a failure state such as a crash in the system or component
NFR5	Performance	High Data Ingestion: The system must be able to successfully process, evaluate and store device data (JSON inputs) of over 40 MB in size
NFR6	Supportability	The system shall be extensible for future technologies (versions of FHIR)
NFR7	Regulatory	General Data Protection Regulation (GDPR) compliance

4.1.4 Domain

Having defined the functions of the platform, the domain and the entities that compose it must be defined and structured. It could be extracted from the requirements the existence of four domain concepts around which the platform will function: Patient, Clinical Trial, Biosensor Data, Sleep+ and Bedroom+ Data and Bundles. For each one of these, a matching FHIR resource that supports them must be assigned.

Regarding the **Patient**, FHIR retains its own definition of a Patient, as it is present among the various FHIR Resource Types, showcased in Appendix A. According to FHIR documentation, a Patient resource can hold personal, professional and clinical data that make up the presence of said patient within a system. Through discussion with stakeholders, only basic identification data for the patient is required for the platform. As such, the FHIR definition of patient can be used, without any customization.

For a **Clinical Trial**, the adoption of FHIR gets more complicated. Among the FHIR Resource Types, no clinical trial resources were found. This, however, does not present a problem, as the purpose of the FHIR Resource Types library is to be taken and molded by developers in order to match the domain requirements of a system in development. As

such, the thing to do is to find among the FHIR Resource Types one that can support the SMART-HEALTH-4-ALL definition of a Clinical Trial. The selected one was Organization (*FHIR Organization 2022*), not only due to its reference relationship with patients, but due to its attributes being beneficial and in accordance with those needed (a name for the trial, a classification type and a description).

When it comes to **Biosensor Data**, it is a similar case to Clinical Trials, where no direct equivalent exists in the FHIR Resource Library, and one of those resources must be modified. The provided information regarding what composes a biosensor data record is in Figure 4.3:

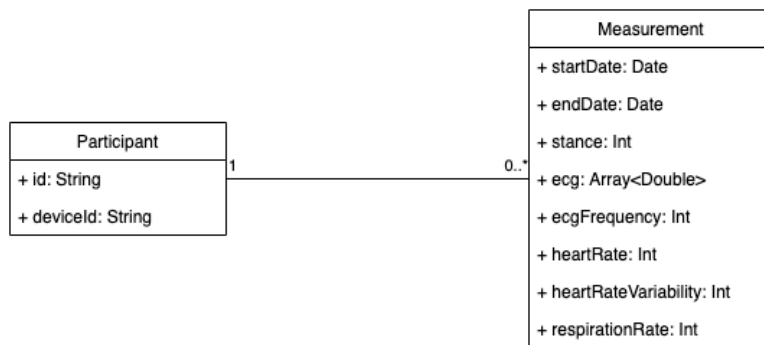


Figure 4.3: Biosensor Data attributes

The chosen resource from FHIR to represent the Biosensor Data was Observations (*FHIR Observation 2022*), and a comparison between the attributes for both is in Table 4.3.

Table 4.3: Biosensor Data Concept and FHIR Observation Comparison

Biosensor Data		FHIR	
Attribute	Description	Attribute	Type
startDate	Start of measurement	effectivePeriod	Period (composed of dateTime "start" and "end" values)
endDate	End of measurement		
stance	Stance the patient was on while taking the vital signs	valueInteger	integer
ecg	Array of ECG values	referenceRange	Array of BackboneElement (elements that can contain various measures, including quantities, age values and even text extracts)
ecgFrequency	frequency of ECG	component	Array of BackboneElement
heartRate	calculated from ECG		
heartRate Variability	calculated from ECG		
respirationRate	calculated from ECG		

Similarly to the Biosensor Data, the **Sleep+ and Bedroom+ Data** results from a series of measurements and needs to be converted to FHIR Observation.

The Sleep+ and Bedroom+ data set starts by identifying the user of the devices (a patient). This is followed by a timestamp of the date the readings were sent and timestamps indicating the start and end of the readings. Finally, there are the Sleep+ measurement (which includes average, maximum and minimum temperature, average, maximum and minimum humidity and three degrees (arc degrees) of head positioning) and the Bedroom+ measurement (average, maximum and minimum temperature and average, maximum and minimum humidity).

The association between the Sleep+ and Bedroom+ data attributes and the FHIR Observation attributes is present in Table 4.4.

Table 4.4: Sleep+ and Bedroom+ Data and FHIR Observation Comparison

Sleep+ and Bedroom+ Data	FHIR	
Attributes	Attribute	Type
start_reading	effectivePeriod	Period
end_reading		
sent_timestamp	issued	Instant
Sleep+ Temperature (Average, Minimum and Maximum)	component	Array of BackboneElement
Sleep+ Humidity (Average, Minimum and Maximum)		
Sleep+ Head Positions (Position 1, Position 2, Position 3)		
Bedroom+ Temperature (Average, Minimum and Maximum)		
Bedroom+ Humidity (Average, Minimum and Maximum)		

Finally, the **Bundle** concept was compatible with the bundle resource (*Bundle 2023*), on which it is based. The resource entails some identification data and then an array of entries, which will be filled with the likes of Patients, Organizations (Clinical Trials) and Observations (Biosensor Data and Sleep+/Bedroom+ Data) previously described.

Taking all of this into consideration, the approach to the domain documentation involved the definition of two domain models: one pertaining to the entities and relationships used

by the Management API and another one depicting the data structuring used by the FHIR server to represent these same entities and relationships.

The first of these is present in Figure 4.4, where it's established the dynamic of a patient "belonging" to a clinical trial and Biosensor and Sleep+/Bedroom+ measurements referencing this same patient.

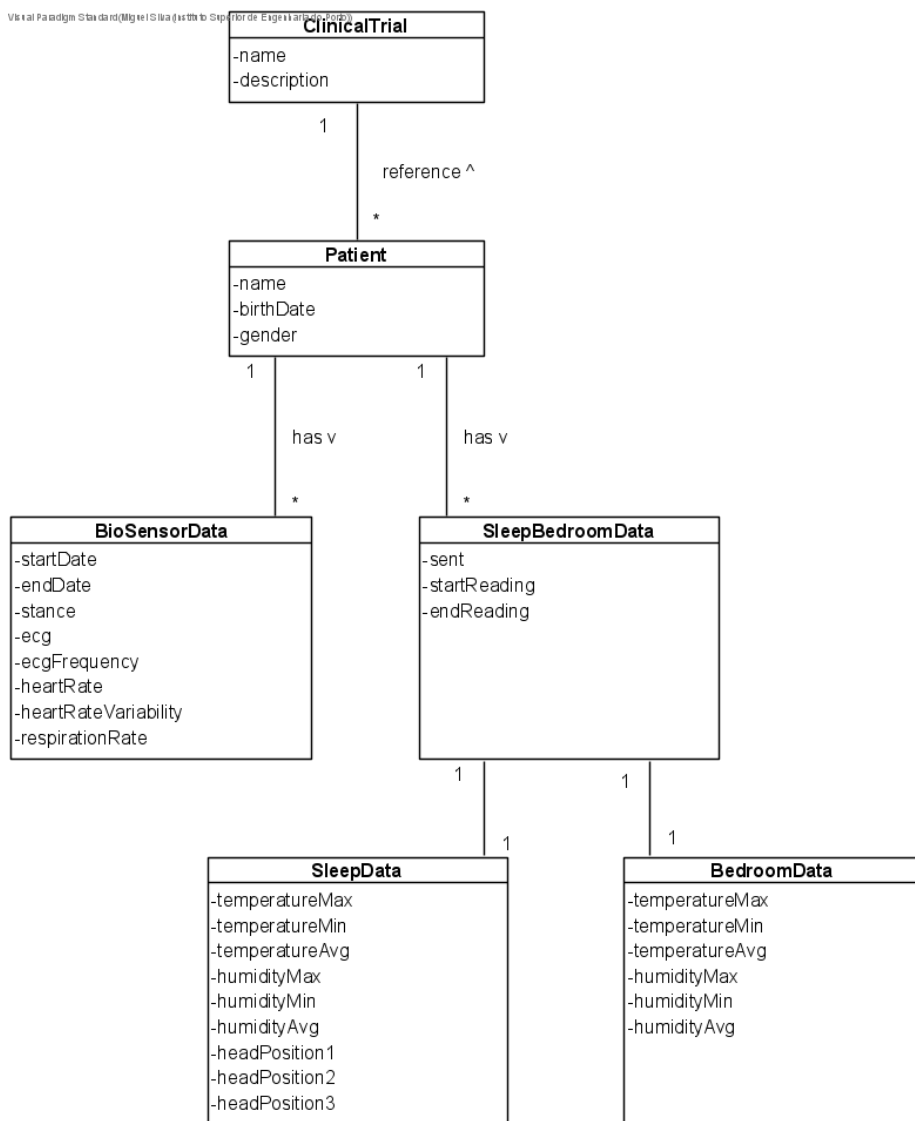


Figure 4.4: Management API Domain Model

The second domain model considers the existing resources and relational rules of HL7 FHIR. With this, we can build a small, direct domain model visualizing the FHIR-inspired entities that the platform will be managing, as seen in Figure 4.5. In it, the details of the Observation, Patient and Organization FHIR resource types, their corresponding entities in the Management API and their possible inclusion in a Bundle, are all depicted.

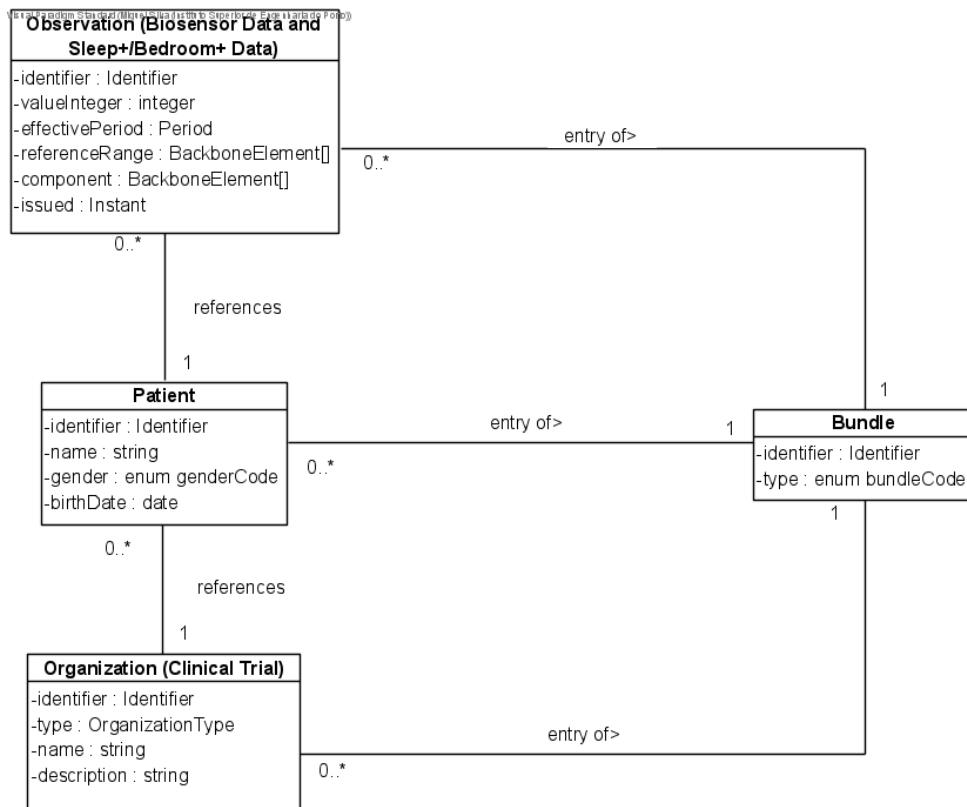


Figure 4.5: FHIR-based Domain Model

4.2 Design

To express the design of the platform, the 4+1 architecture view model was applied, exploring the logical view (described during the analysis on the previous chapter), the development view (through component diagrams), the physical view (with deployment diagrams) and the process view (using sequence diagrams).

4.2.1 The 4+1 View Model

The 4+1 Architectural View Model was created by Kruchten (1995) and aims to describe software architecture by presenting it "[...] composed of multiple views or perspectives". These encompass:

- The logical view, the model object of the design, and "[...] supports the functional requirements — what the system should provide in terms of services to its users [...] taken (mostly) from the problem domain, in the form of objects or object classes";
- the process view, which captures the concurrency and synchronization aspects of the design. Among its utilities is showing "[...] how the main abstractions from the logical view fit within the process architecture — on which thread of control is an operation for an object actually executed";
- the development view, which describes the static organization of the software in its development environment. "It is the basis for establishing a line-of-product";

- the physical view, which describes the mapping(s) of the software onto the hardware and reflects its distributed aspect;

By presenting the architecture through these views, analysis of resources, costs and planning become more manageable. Besides this, it also allows various stakeholders to find what they want to know about the software architecture with more ease.

4.2.2 Platform Architecture

Logical View

The logical view is expressed in Chapters 4.1, where the analysis looks at the functionality of the platform through a Use Case Diagram and a Domain Model.

Process View

To express this view, we can take a look at two requirements identified in Chapter 4.1.2 and view each of them through the perspective of a System Sequence Diagram (SSD).

Starting with the FR4 Requirement - Incorporate Biosensor Data - it consists of the platform receiving data pertaining measurements made by biosensors to a specific patient, within a period of time. The diagram in Figure 4.6 sees the flow of information, starting with the emitter providing the information (can be a device, a service or an application and is controlled by the Health Professional), which is received by the management API, specifically its data controller, before being sent to the data service, where the data is restructured into the JSON format expected by the FHIR Server. The restful server receives the restructured data, invokes the resource provider for the resource corresponding to biosensor data (observation, as explained in chapter 4.1.4), which deals with the persistence of it.

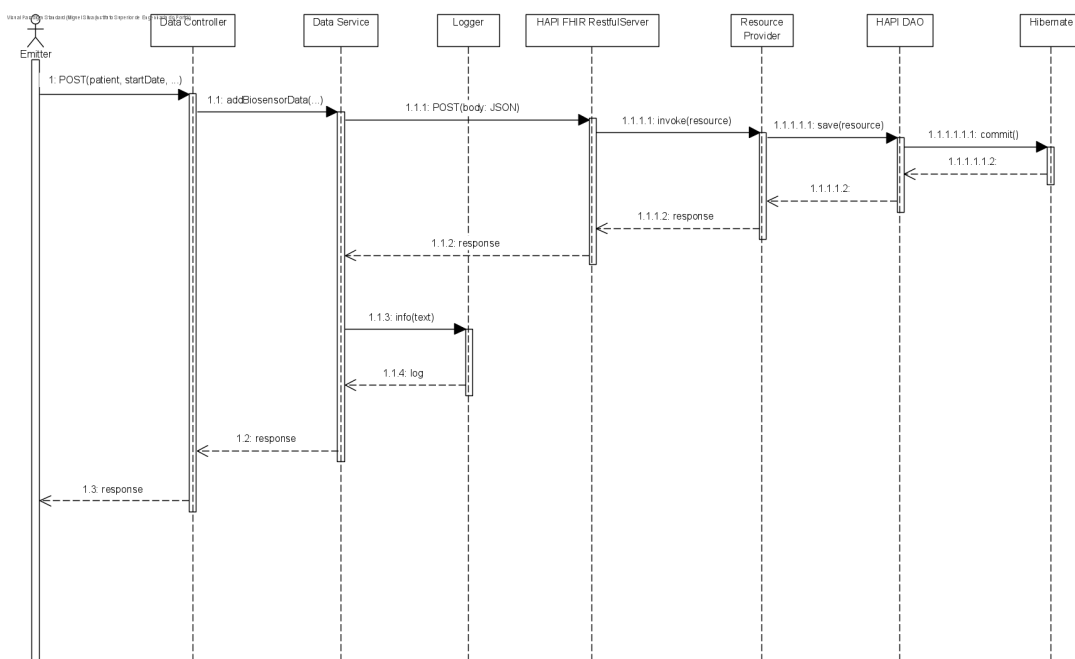


Figure 4.6: SSD for Use Case of requirement FR4

Once the persistence of the Biosensor data is finished and the DataService is aware of this, it proceeds with the logging of the action, structuring a logging message that includes all the relevant data.

The same logic of this previous SSD can be applied to requirement FR5, with the difference being that, since a bundle will contain multiple resources and not just one, there is a loop happening pertaining to an access of multiple resource providers, for each of the resource entries that make up the bundle, as can be seen in Figure 4.7:

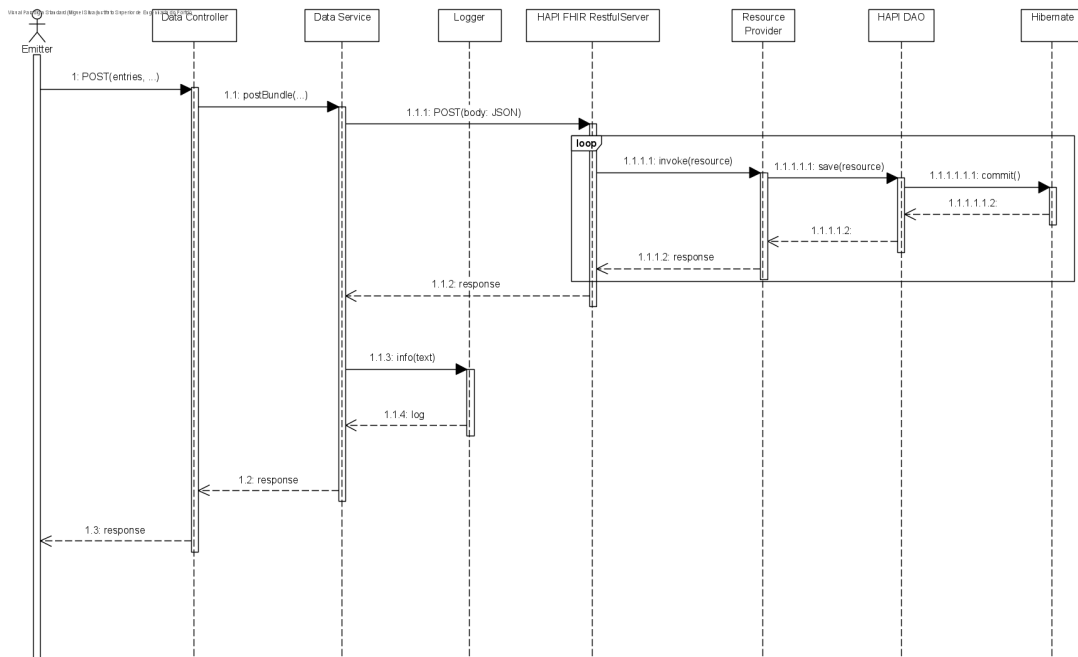


Figure 4.7: SSD for Use Case of requirement FR5

Development View

The platform was conceptualized as being part of a testing environment that includes devices, services and applications, as well as combinations of those three. As such, it was important to design these environments through component diagrams, making up the development view of the PPS1 architecture. Initially, a more generic point of view of the environment will be presented, before going over these possible combinations and discussing some of the specifics that comprise them.

As can be seen in Figure 4.8, there is a **general testing scenario**, which depicts a System Under Tests (these tests being another term for clinical trials), using User Interface (UI) and service interactions to communicate with the platform which will store its data and run tests on it. The purpose for this is to ensure that both the front-end and the back-end of the system, if the system has a division of these ends (for example, an application), will be part of the test. For situations where only the back-end of a system (such as a standalone service), only the service interaction is necessary for the testing.

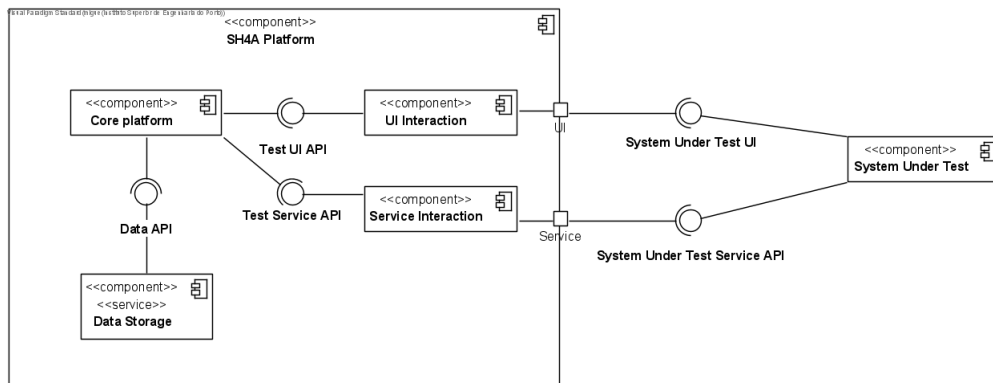


Figure 4.8: Development View - General testing scenario

Looking with more detail at these scenarios, we start by seeing one where a **service is being tested** during its clinical trial process of development. A diagram representing this scenario can be seen in Figure 4.9. In it, the service being tested interacts with the test orchestration which will make use of the platform and management services to store and evaluate the data from the service.

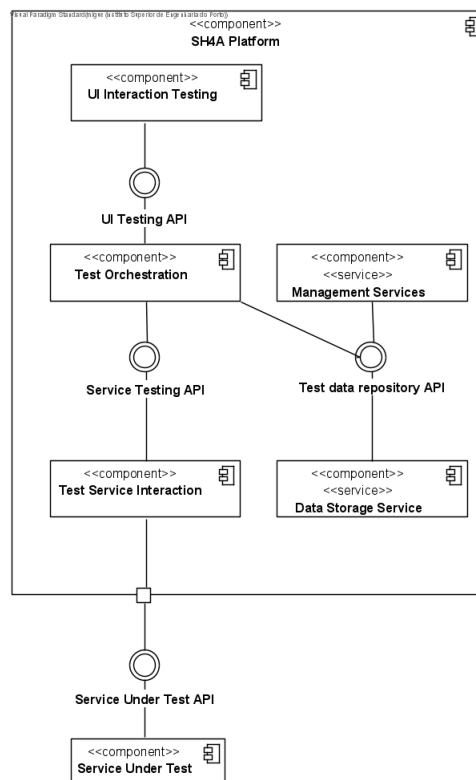


Figure 4.9: Development View - Test of a Service

For the testing of a device, this may involve the involvement of services and/or applications, which will be addressed later, or the **device being tested on its own**. For this, there is the need to mock/simulate a service. This is visible in Figure 4.10, where the device connects to the mocked service (the Test Stub) which will provide a middleman between the device and the test orchestration as we know it.

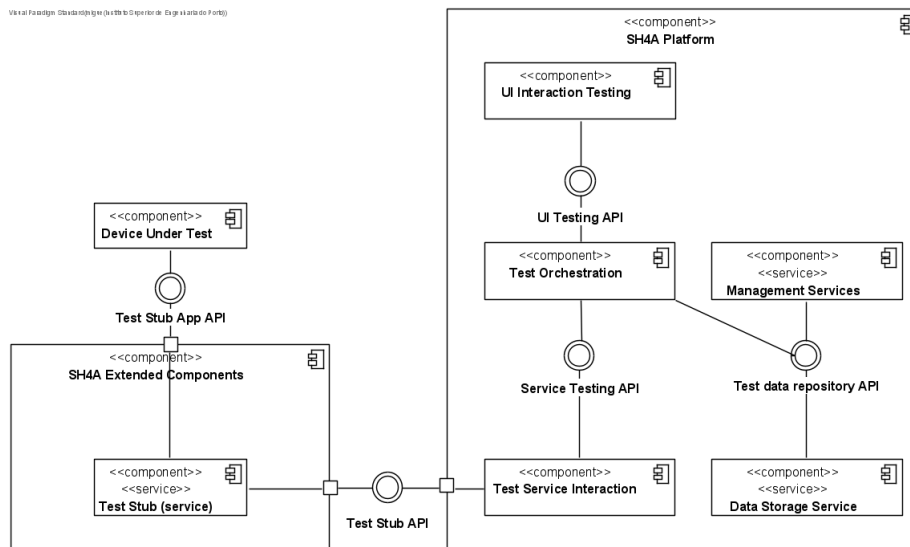


Figure 4.10: Development View - Test of a Device

If a **device requires user interaction**, then it is assumed there is a User Interface that also needs testing, and as such, the previous diagram must be completed with a UI Interaction testing, as in Figure 4.11:

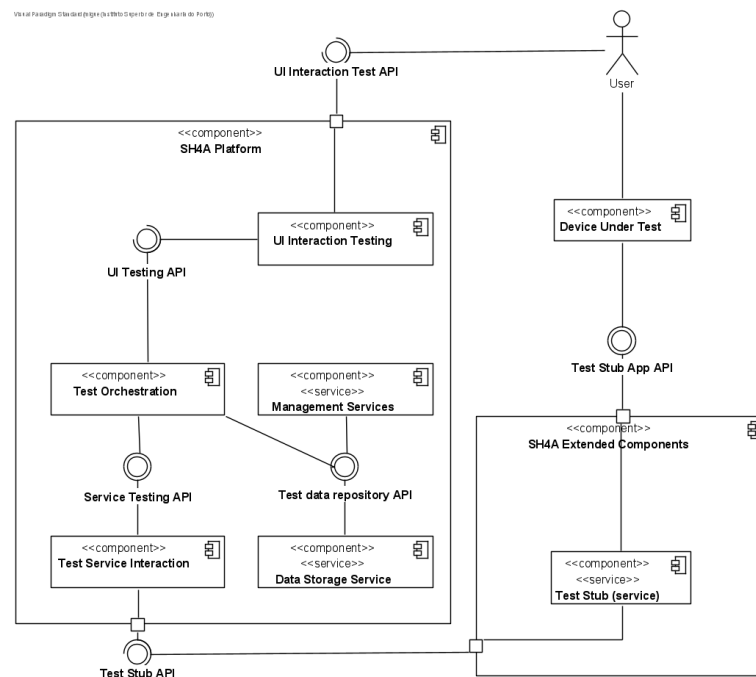


Figure 4.11: Development View - Test of a Device with User Interaction

The same logic as this last scenario works for **applications that are being tested**, as their interface and their services need testing, much as in Figure 4.12, where the app under test connects to the UI testing and a service test stub for its service testing.

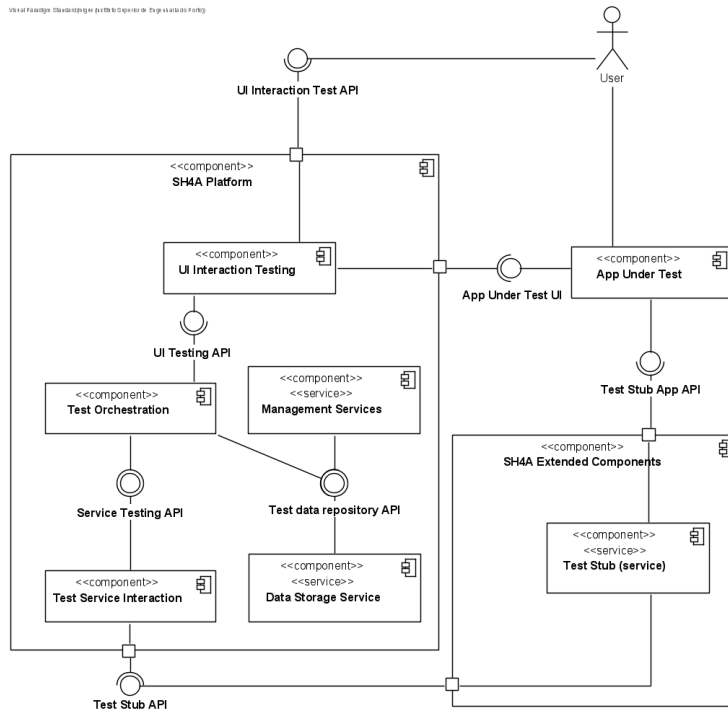


Figure 4.12: Development View - Test of an Application

Then, combinations between applications, devices and services in test can be designed as well, such as those present in Figure 4.13, which sees an **application in testing and a service in testing** cooperating to feed the test orchestration with data, and Figure 4.14, which sees the same logic, but with the **cooperation between a device and a service**, both in testing.

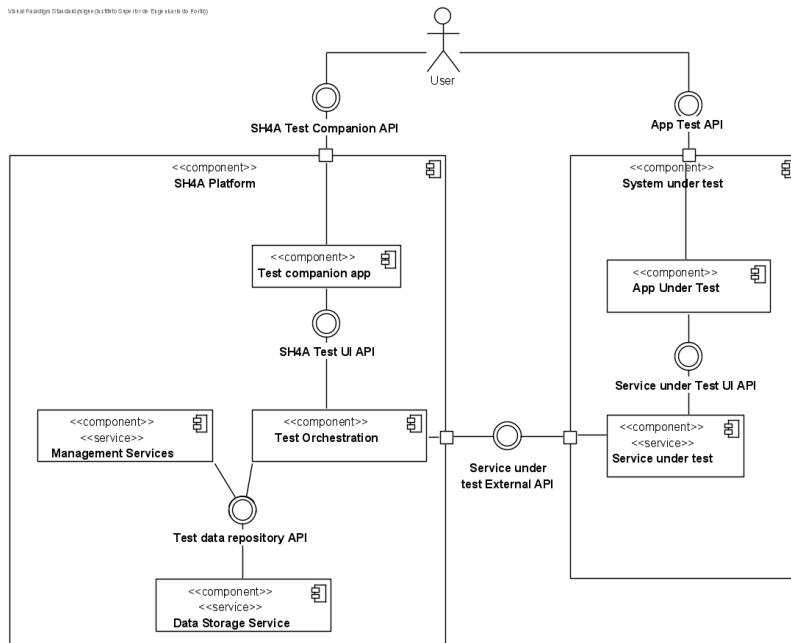


Figure 4.13: Development View - Test of a system App + Service(s)

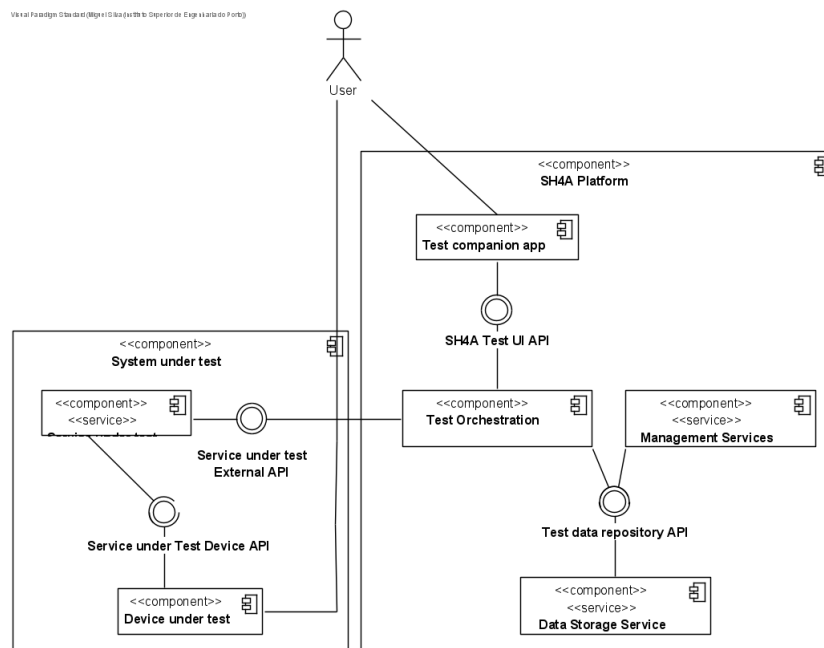


Figure 4.14: Development View - Test of a system Service(s) + Device(s)

For **monolithic systems**, defined as systems where applications, services and devices are all being tested, a setup like the one displayed in Figure 4.15 is ideal. The user is testing applications and devices, both connecting to services that will feed the test orchestration.

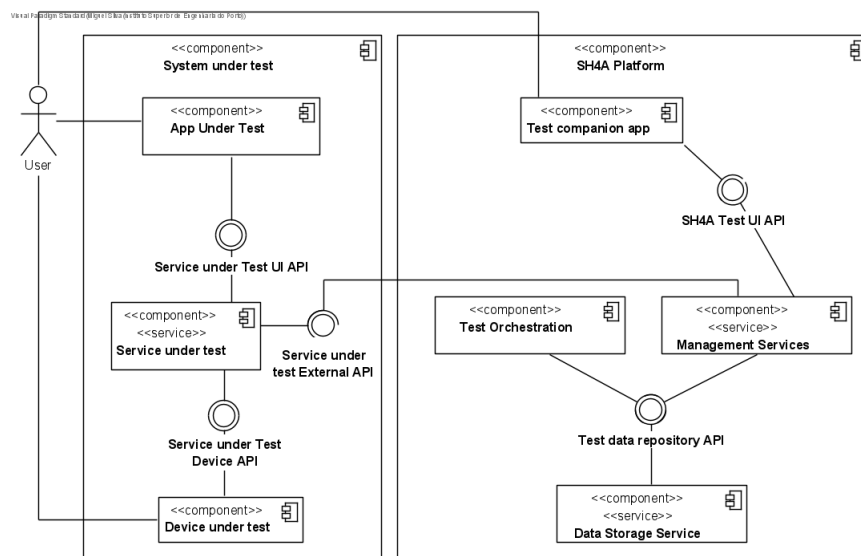


Figure 4.15: Development View - Test of a monolithic system (App + Service(s) + Device(s)) with user interaction with devices

Additionally, variations of monolithic systems were considered, such as a **monolithic system where the user is only connected to applications** and not devices (4.16) and a **monolithic system where the applications and devices connect directly**, as opposed to having a service as a middleman (4.17).

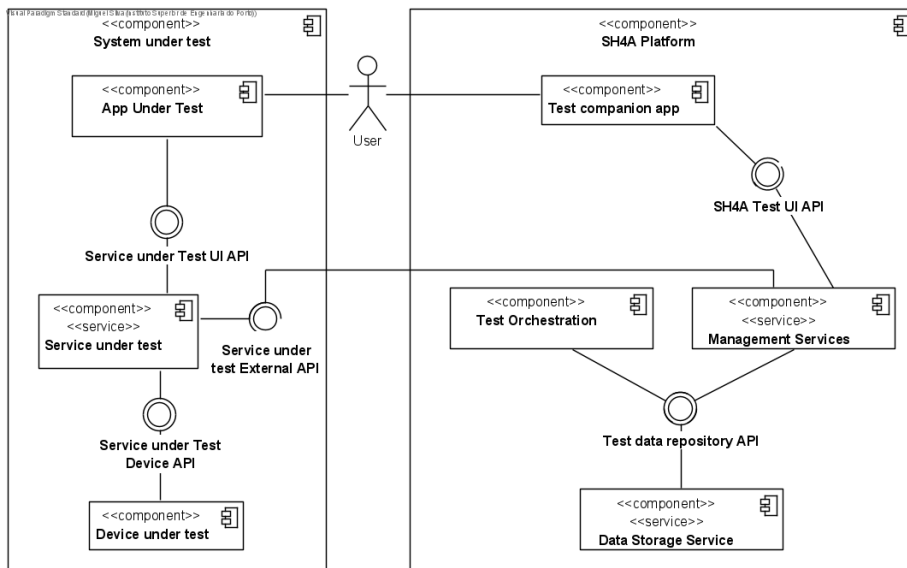


Figure 4.16: Development View - Test of a monolithic system without user interaction with the devices

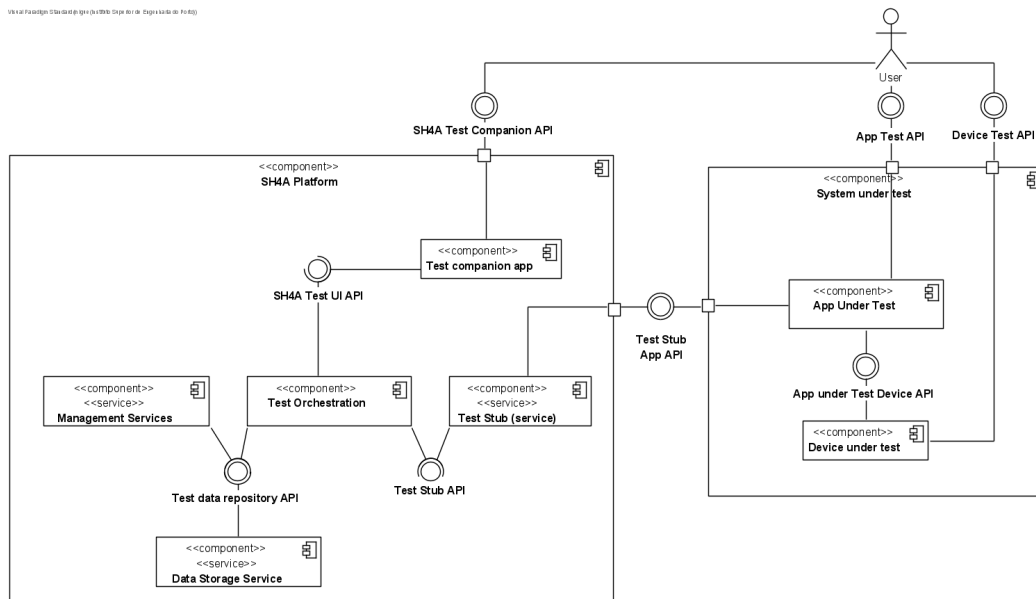


Figure 4.17: Development View - Test of monolithic system with devices connected to App

Physical View

For this view, the connections between the various servers, APIs and physical components were summarized using a deployment diagram, that shows how these communicate (through HTTP/HTTPS protocols).

In Figure 4.18, the user environment, composed of, as far as we know, a computer device, can access the testing environment, which houses the system(s) under test (either to control them or to oversee their performance) and also access the SH4ALL platform, containing

the software for the test orchestration, the platform that will bridge the tests and the persistence using FHIR and the log file(s). The SH4ALL platform is communicating with the HAPI FHIR server, containing the database and a interface for the database management.

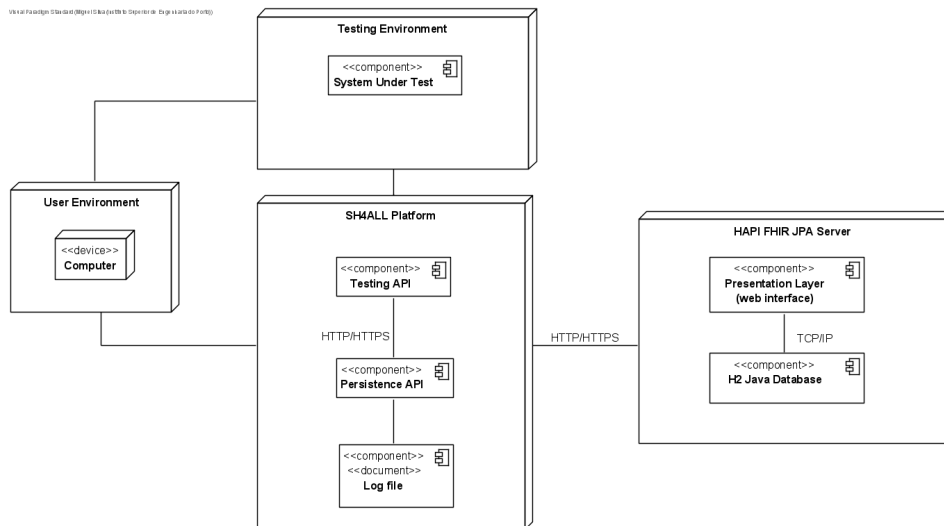


Figure 4.18: Physical View - Deployment diagram

Chapter 5

Implementation

In this chapter, technical decisions, such as programming languages, frameworks and platforms to be used, in the implementation of the solution will be firstly addressed, followed by a detailed guide to the setup and execution of the two key components of the platform envisioned in the setup: the FHIR Server and its Management application.

5.1 Technical Decisions

As showcased, for instance, in Figure 4.9, the platform will be composed of an API and a storage service. The API will be responsible for communicating with the technology (device, application or service) under trial, receiving its data and setting it up so that the storage service can interpret and restructure it according to the FHIR standard, leaving it to the API to return the restructured information.

For the API, the chosen programming language for its development was Java, utilizing the *Spring Framework* (2023), with the Spring Boot extension, as it stood as a direct, simpler and more cost, resource and time effective option, in comparison to other frameworks that were briefly considered.

As for the FHIR server, there were two possible options to choose from (*FHIR Server Types* 2023), differing in complexity, variety of functions and approaches to data storage and security:

- **Plain Server** - described as "[...] an implementation of a FHIR server against an arbitrary backend", it consisted of a server capable of HTTP Processing, Parsing, Serialization and managing FHIR REST semantics, while requiring manual backend configurations (these included servlet setups and the definition of key complex classes, such as Resource Providers);
- **JPA Server** - is "[...] a complete implementation of a FHIR server against a relational database", which means that it consists of the plain server packaged with multiple classes, such as Resource Providers and Plain Providers, setup in a default, fully runnable way, providing its own database schema and handling all storage and retrieval logic, letting developers make modifications and molding the server to meet the needs of its implementation. The JPA Server also comes with a built-in interface that facilitates the understanding of its schemas and semantics, as well as the requests its is programmed to receive;

Having considered both options, the **JPA Server** was chosen due to it fulfilling a lot of the coding and configuration needed, allowing for more focus to be put in the API and the management of the incoming and outgoing data in it.

5.2 Setup of HAPI FHIR Server

The *HAPI FHIR JPA Server* (2023) is executable as is, in its starter state provided by HL7, and presents an architecture based around **Resource Provider** classes, which exist for each resource type in the FHIR semantic library, as can be seen in Figure 5.1:

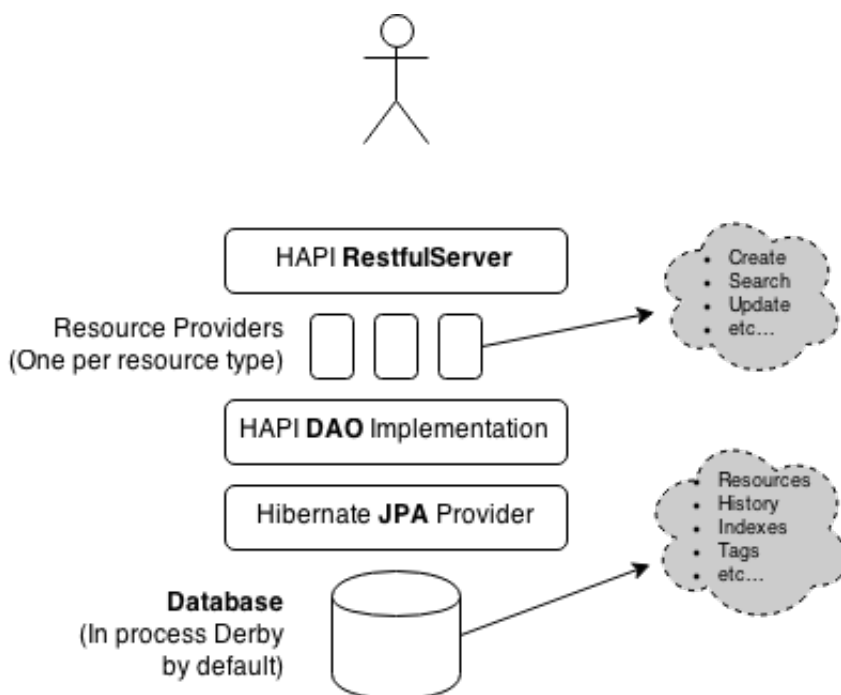


Figure 5.1: HAPI FHIR JPA Server Architecture (*FHIR Server Types 2023*)

Resource Providers contain the business logic of the server, defining which resource type from the overarching FHIR Semantic Library is considered by the server and which functions can be executed in relation to that resource.

As such, the Resource Provider will have the possibility of defining CRUD methods, while having at least one obligatory **Search** method.

In Listing 5.1, an example of a Resource Provider for the patient concept supported by FHIR can be seen. This class starts by having a method responsible for identifying the resource type it manages (lines 13 to 16). Afterwards, two Search methods are present, one returning all the patients in the database (lines 18 to 23) and the other finding and returning in a list one or more patients with a specific family name (lines 25 to 37). The final two methods are a Read method which finds a specific patient through their identification value (lines 39 to 46) and a Create method that receives the raw data of a patient, sets a new id for them and persists the data (lines 48 to 55).

```
1 package ca.uhn.fhir.example;
2
3 import ca.uhn.fhir.rest.annotation.IdParam;
4 import ca.uhn.fhir.rest.annotation.Read;
5 import ca.uhn.fhir.rest.server.IResourceProvider;
6 import ca.uhn.fhir.rest.server.exceptions.ResourceNotFoundException;
7 import org.hl7.fhir.r4.model.IdType;
8 import org.hl7.fhir.r4.model.Patient;
9 import org.hl7.fhir.instance.model.api.IBaseResource;
10
11 public class RestfulPatientResourceProvider implements IResourceProvider
12     {
13     @Override
14     public Class<? extends IBaseResource> getResourceType() {
15         return Patient.class;
16     }
17
18     @Search
19     public List<Patient> search() {
20         List<Patient> retVal = new ArrayList<Patient>();
21         retVal.addAll(myPatients.values());
22         return retVal;
23     }
24
25     @Search
26     public List<Patient> search(@RequiredParam(name = Patient.SP_FAMILY)
27         StringParam theParam) {
28         List<Patient> retVal = new ArrayList<Patient>();
29         for (Patient next : myPatients.values()) {
30             String familyName = next.getNameFirstRep().getFamily().toLowerCase();
31             if (familyName.contains(theParam.getValue().toLowerCase()) ==
32                 false) {
33                 continue;
34             }
35             retVal.add(next);
36         }
37         return retVal;
38     }
39
40     @Read
41     public Patient read(@IdParam IdType theId) {
42         Patient retVal = myPatients.get(theId.getIdPart());
43         if (retVal == null) {
44             throw new ResourceNotFoundException(theId);
45         }
46         return retVal;
47     }
48
49     @Create
50     public MethodOutcome create(@ResourceParam Patient thePatient) {
51         int id = myNextId++;
52         thePatient.setId(new IdType(id));
53         myPatients.put(Integer.toString(id), thePatient);
54         return new MethodOutcome().setId(thePatient.getIdElement());
55     }
56 }
```

Listing 5.1: Resource Provider Example

Other than the annotations presented in the previous example (@Search, @Read, @Create), there are others to keep in mind while developing a Resource Provider, that pertain to a variety of situations:

- **@Update** - Fully replaces a specific resource using an identification value and an object it receives as a parameter, using the @ResourceParam, which contains the new resource data;
- **@Delete** - Locates a specific resource using its identification value and removes it from the database;
- **@Patch** - Replaces a property from a specific resource using an identification value and a @ResourceParam containing the new property data;
- **@Validate** - used to test if a resource respects the business logic and is passable to be saved in the server;
- **@History** - these methods will provide a complete overview of resources and their versions over time, depending on the specific sub-category of History request that is used:
 - An **Instance History** will receive an identification value and a resource type indicator to return the various versions of a specific resource (for example, to check the version history of a specific patient).
 - A **Type History** will receive a resource type indicator and return all the versions of all the resources of that type (for example, all the version histories of all the patients).
 - The **Server History** is the heaviest History procedure, not receiving any parameters, but returning the version histories of every single resource of every single type in the database;
- **@Transaction** - used in the processing of a bundle of resources, receiving a large object in its parameters, containing multiple resources and the type of action that each requires (to be create, updated, read, deleted, among others), allowing for a considerable number of processes to be executed, sequentially, in a single, atomic operation;
- **@Operation** - special annotations that grant methods a functionality akin to a Remote Procedure Call (RPC). Essentially, flagging methods to be used when a specific "\$name" convention is used. For instance, if a method is annotated with @Operation(name="\$everything") in a Resource Provider for the Patient resource, this means that said method will be put to use when the Uniform Resource Locator (URL) containing "/Patient/\$everything" is invoked. These annotations prove useful when none of the other fit the intended purpose of an operation that the Resource Provider must execute;
- **@Destroy** - defines the steps the resource provider must automatically take in case the server it is connected to shuts down or if any other circumstance calls for its stoppage;

Any and all of the defined Resource Providers will be associated to a restful server, as shown in the architecture of Figure 5.1. This happens during the initialization of the servlets, as can be seen in Listing 5.2. In it, an exemplary Restful Servlet is flagged using the **@WebServlet** annotation, which sets the URL pattern that requests made to it must follow, along with a display name (line 1), while the previously described Resource Provider for patients, as well as one other Resource Provider for Observations, are added to the servlet's list of providers (lines 11 to 14):

```
1 @WebServlet(urlPatterns= {"/fhir/*"}, displayName="FHIR Server")
2 public class ExampleRestfulServlet extends RestfulServer {
3
4     private static final long serialVersionUID = 1L;
5
6
7
8     @Override
9     protected void initialize() throws ServletException {
10         List<IResourceProvider> resourceProviders = new ArrayList<
11         IResourceProvider>();
12         resourceProviders.add(new RestfulPatientResourceProvider());
13         resourceProviders.add(new RestfulObservationResourceProvider());
14         (...)
15         setResourceProviders(resourceProviders);
16     }
17 }
18 }
```

Listing 5.2: RestfulServlet Example

5.3 Setup of Management API

For this section, the first aspects to consider are the model classes, according to the needs of the stakeholders. Afterwards, there are the services that manage the interactions with the server and the controllers that receive and process external requests. Their setup and specifications will all be addressed using code examples and extracts.

5.3.1 Models

As an example of the model classes needed, we have the more complex cases of the **Biosensor Data** and the **Sleep+/Bedroom+** Data.

According to the information provided to the development team (showcased in Figure 4.3), the Biosensor Data that the Smart Health technology captures includes a reference to a specific patient (through the use of an identification value for that same patient). Beyond that, it states two dates indicating the start and end of the measurement, a number indicating the stance of the patient during the procedure, multiple ECG measurements, as well as other numeric values, such as the ECG Frequency, the patient's heart rate, the variability of the heart rate and the patient's respiration rate.

As such, a model class was created to embody the received data, which can be seen in Listing 5.3. In it, the model class defines the properties mentioned (lines 4 to 13). This is followed by the required empty constructor (lines 16 and 17) and a complete constructor (lines 20 to 30). Afterwards, the getter and setter methods for each property are added, as exemplified by the getter and setter for the startDate property (lines 33 to 40).

```
1 public class BioSensorData {
2
3
4     public String id;
5     public String participantId;
6     public Date startDate;
7     public Date endDate;
8     public int stance;
9     public double[] ecg;
10    public int ecgFrequency;
11    public int heartRate;
12    public int heartRateVariability;
13    public int respirationRate;
14
15
16    public BioSensorData(){
17    }
18
19
20    public BioSensorData(String participantId, Date startDate, Date
21    endDate, int stance, double[] ecg, int ecgFrequency, int heartRate,
22    int heartRateVariability, int respirationRate){
23        this.participantId = participantId;
24        this.startDate = startDate;
25        this.endDate = endDate;
26        this.stance = stance;
27        this.ecg = ecg;
28        this.ecgFrequency = ecgFrequency;
29        this.heartRate = heartRate;
30        this.heartRateVariability = heartRateVariability;
31        this.respirationRate = respirationRate;
32    }
33
34    public Date getStartDate() {
35        return this.startDate;
36    }
37
38    public void setStartDate(Date startDate) {
39        this.startDate = startDate;
40    }
41
42    (...)
43
```

Listing 5.3: BiosensorData Model Class

Regarding the Sleep+/Bedroom+ Data, as defined in Table 4.4, for this class, which is showcased in Listing 5.4, we must first define the connection to a specific patient, through a user identification (line 5), followed by the timestamps marking the start and end of the

reading and the moment the reading was sent to the platform (lines 6 to 11). Once this is done, the Sleep and Bedroom measurement values are set, using Value Objects **SleepData** and **BedroomData** respectively (lines 12 and 13).

Much like the previous example, the definition of the properties is followed by an empty constructor (lines 16 and 17), a complete constructor (lines 20 to 27), and the getter and setter methods (example in lines 30 to 37).

```
1 public class SleepBedroomData {
2
3
4     public String id;
5     public String userId;
6     @JsonFormat(shape=JsonFormat.Shape.STRING, pattern="yyyy-MM-dd HH:mm
:ss", timezone="Europe/Lisbon")
7     public Timestamp sent;
8     @JsonFormat(shape=JsonFormat.Shape.STRING, pattern="yyyy-MM-dd HH:mm
:ss", timezone="Europe/Lisbon")
9     public Timestamp startReading;
10    @JsonFormat(shape=JsonFormat.Shape.STRING, pattern="yyyy-MM-dd HH:mm
:ss", timezone="Europe/Lisbon")
11    public Timestamp endReading;
12    public SleepData sleepPlus;
13    public BedroomData bedroomPlus;
14
15
16    public SleepBedroomData() {
17    }
18
19
20    public SleepBedroomData(String userId, Timestamp sent, Timestamp
startReading, Timestamp endReading, SleepData sleepPlus, BedroomData
bedroomPlus) {
21        this.userId = userId;
22        this.sent = sent;
23        this.startReading = startReading;
24        this.endReading = endReading;
25        this.sleepPlus = sleepPlus;
26        this.bedroomPlus = bedroomPlus;
27    }
28
29
30    public Timestamp getSent() {
31        return this.sent;
32    }
33
34
35    public void setSent(Timestamp sent) {
36        this.sent = sent;
37    }
38
39
40    (...)
```

Listing 5.4: SleepBedroomData Model Class

The Sleep Data object should contain all the relevant data to sleep measurements, which includes **Temperature**, **Humidity** and **HeadPosition**, while the Bedroom Data should have the necessary data for the bedroom measurements, which are just **Temperature** and **Humidity** values. These two objects can be viewed in Listings 5.5 and 5.6, respectively.

```
1 public class SleepData {
2
3     public Temperature temperature;
4     public Humidity humidity;
5     public HeadPosition headPosition;
6
7     public SleepData() {
8     }
9
10    public SleepData(Temperature tem, Humidity hum, HeadPosition hp) {
11        this.temperature = tem;
12        this.humidity = hum;
13        this.headPosition = hp;
14    }
15
16    public Temperature getTemperature() {
17        return this.temperature;
18    }
19
20    public void setTemperature(Temperature temperature) {
21        this.temperature = temperature;
22    }
23
24    (...)
```

Listing 5.5: SleepData Class

```
1 public class BedroomData {
2
3     public Temperature temperature;
4     public Humidity humidity;
5
6     public BedroomData() {
7     }
8
9     public BedroomData(Temperature tem, Humidity hum) {
10        this.temperature = tem;
11        this.humidity = hum;
12    }
13
14    public Temperature getTemperature() {
15        return this.temperature;
16    }
17
18    public void setTemperature(Temperature temperature) {
19        this.temperature = temperature;
20    }
21
22    (...)
```

Listing 5.6: BedroomData Class

A **Temperature** measurement is set as the maximum, minimum and average temperature values. The same logic applies to a **Humidity** measurement. Both of these can be seen in Listing 5.7 and 5.8, respectively.

```
1 public class Temperature {
2
3
4     public double average;
5     public double min;
6     public double max;
7
8
9     public Temperature() {
10    }
11
12
13    public Temperature(double averageVal, double minimum, double maximum
14    ) {
15        this.average = averageVal;
16        this.min = minimum;
17        this.max = maximum;
18    }
19
20    (...)
```

Listing 5.7: Temperature Class

```
1 public class Humidity {
2
3
4     public int average;
5     public int min;
6     public int max;
7
8
9     public Humidity() {
10    }
11
12
13    public Humidity(int averageVal, int minimum, int maximum) {
14        this.average = averageVal;
15        this.min = minimum;
16        this.max = maximum;
17    }
18
19
20    (...)
```

Listing 5.8: Humidity Class

A **Head Position** measurement is categorized by three integer values, referring to three different positions held by the patient, as set in Listing 5.9.

```
1 public class HeadPosition {
2
3
4     public int position1;
5     public int position2;
6     public int position3;
7
8
9     public HeadPosition() {
10    }
11
12
13    public HeadPosition(int position1, int position2, int position3) {
14        this.position1 = position1;
15        this.position2 = position2;
16        this.position3 = position3;
17    }
18
19    (...)
20
```

Listing 5.9: HeadPosition Class

5.3.2 Service

A Data Service, responsible for communicating with the FHIR Server, must be developed in order to implement the various use cases listed in section 4.1.2, while also considering some of the non-functional aspects mentioned in section 4.1.3.

Firstly, we will setup a **Logger** object, to be used by the service, so that every action taken by the Management API is registered in a log file, which can be accessed anytime to review the functionality of the system. This setup is visible in Listing 5.10.

```
1 import org.springframework.stereotype.Component;
2
3 import org.slf4j.Logger;
4 import org.slf4j.LoggerFactory;
5
6 @Component
7 public class DataService {
8
9     Logger logger = LoggerFactory.getLogger(DataService.class);
10
11    (...)

```

Listing 5.10: Data Service Logger

The first use case, **UC1**, entailed the creation of Clinical Trials in the FHIR Server. As previously described, the FHIR Resource Type that will be used to represent trials in the server will be the **Organization** resource. The Data Service must prepare a request that considers this.

That approach is present in Listing 5.11, where the URL for the Organization Resource Provider in the FHIR Server (line 3) is used by an Hypertext Transfer Protocol (HTTP) Client (lines 5 and 7). In the request body, it is specified the intention of persisting an organization resource (line 9), while headers indicating the exchange of data in a JSON format

(lines 15 and 16) finalize the setup of the request, which is afterwards executed (line 20). The response of the request, if successful, will be extracted (line 22) and logged into the log file (lines 24 to 26), before being properly returned by the method (line 28).

```
1 public String createTrial() throws ClientProtocolException, IOException,
2     ParseException {
3     String url = "http://localhost:8080/fhir/Organization";
4     CloseableHttpClient httpClient = HttpClients.createDefault();
5     HttpPost httpPost = new HttpPost(url);
6     String body = "{\"resourceType\":\"Organization\"}";
7     StringEntity params = new StringEntity(body);
8     params.setContentType("application/json");
9     httpPost.addHeader("content-type", "application/json");
10    httpPost.addHeader("accept", "application/json");
11    httpPost.setEntity(params);
12    CloseableHttpResponse response = httpClient.execute(httpPost);
13    String res = EntityUtils.toString(response.getEntity(), "UTF-8");
14    ;
15    logger.info("\n===== Response to the creation of a Clinical
16    Trial =====");
17    logger.info(res);
18    logger.info("\n=====");
19    ;
20    return res;
21 }
22
23
24
25
26
27
28
29
30
```

Listing 5.11: Data Service Clinical Trial Creation

The next two use cases, **UC2** and **UC3**, relate to the creation of a patient in the FHIR Server and their association to an existing clinical trial. These two were implemented together in a single method, which has the data service following the same steps as the method for Clinical Trial Creation that was previously described, with two major differences: the first is that the resource type this time around is the FHIR Patient; the second is that the request body will require the identification of the clinical trial (perceived by the FHIR Server as an Organization) that the patient is part of.

As such, the method was developed as shown in Listing 5.12, where the identification of the trial is received in the parameters (line 1) and used in the setup of the body (line 9). The rest of the method follows the same logic as the one previously described for the Clinical Trial, with the setup of headers, the execution of the request and the extraction, logging and return of the response.


```

1  public String createPatient(String org) throws
2  ClientProtocolException, IOException, ParseException {
3
4      String url = "http://localhost:8080/fhir/Patient";
5
6      CloseableHttpClient httpClient = HttpClients.createDefault();
7
8      HttpPost httpPost = new HttpPost(url);
9
10     String body = "{\"resourceType\":\"Patient\", \"
11     managingOrganization\": {\"reference\": \"Organization/\" + org +
12     \"}}";
13
14     StringEntity params = new StringEntity(body);
15
16     params.setContentType("application/json");
17
18     httpPost.addHeader("content-type", "application/json");
19     httpPost.addHeader("accept", "application/json");
20
21     httpPost.setEntity(params);
22
23     CloseableHttpResponse response = httpClient.execute(httpPost);
24
25     String res = EntityUtils.toString(response.getEntity(), "UTF-8")
26     ;
27
28     logger.info("\n===== Response to the creation of a Patient
29     =====");
30     logger.info(res);
31     logger.info("\n=====");
32
33     ;
34
35     return res;
36
37 }

```

Listing 5.12: Data Service Patient Creation

Regarding **UC4**, the persistence of Biosensor Data follows the same logic as the previous cases (setup and execution of HTTP POST request followed by extraction, logging and return of response), but presents a much more complex setup of the body of the request, due to the nature of what the FHIR Server expects.

To better understand this, an overview of the JSON body structure needed for a FHIR **Observation**, which is the FHIR Resource Type that is used to represent Biosensor Data (as well as Sleep+Bedroom+ Data) in the FHIR Server, can be seen in Listing 5.13. In it, it is clearly visible the rigid and specific structure and rules that needs to be followed in order to successfully persist the required information.

The correspondence between the properties of the Biosensor Data and the Observation Resource Type will follow the logic described in section 4.1.4 and the conversion between data structures will be executed by the data service.

```
1 {
2   "resourceType": "Observation",
3   "subject": {
4     "reference": "XYZ"
5   },
6   "effectivePeriod": {
7     "start": "yyyy-MM-dd HH:mm:ss",
8     "end": "yyyy-MM-dd HH:mm:ss"
9   },
10  "valueInteger": 123,
11  "referenceRange": [
12    {
13      "high": 123
14    }
15  ],
16  "issued": "yyyy-MM-dd HH:mm:ss",
17  "component": [
18    {
19      "code": {
20        "text": "XYZ"
21      },
22      "valueQuantity": {
23        "value": 123
24      }
25    }
26  ],
27  "code": "XYZ"
28 }
```

Listing 5.13: JSON Body for FHIR Observation

The method in Listing 5.14 handles the persistence of the Biosensor Data, which it receives via its parameters (line 1). After setting up the URL to point to the Observation Resource Provider (line 3), and starting the HTTP Client (lines 5 and 6), the body will be set up.

The first thing to do is create an array of JSON Objects, each one pertaining to an **ECG** measurement of the BiosensorData ECG array (lines 8 to 11). This array will be later put into the request body using the name of "referenceRange" (line 30).

Afterwards, another JSON Array is created, and in it will be inserted the Biosensor Data properties of **ECG Frequency**, **Heart Rate**, **Heart Rate Variability** and **Respiration Rate** (lines 13 to 21, shortened for brevity). For each one of these four properties, a JSON Object is created and in it inserted two other JSON Objects, one containing the name of the property using the designation "text" and another containing the value of the property using the designation "valueQuantity". The four JSON Objects pertaining to the four properties will be added to the JSON Array, which in turn will be added to the body with the name of "component" (line 31).

Finally, the remaining properties of BiosensorData will be directly inserted into the body of the request, with the **id of the patient** the data measurements belong to being inserted with the name of "subject" (line 27), the **start** and **end** timestamps being formatted and inserted as a single JSON Object with the name "effectivePeriod" (line 28) and the **stance** indicator being inserted as "valueInteger" (line 29).

Also, to the body are added the identification of the resource type as Observation (line 26) and a "code" indicating that this particular Observation is a BiosensorData (line 32).

```

1 public String createBiosensorData(BioSensorData bsd) throws
  ClientProtocolException, IOException, ParseException {
2
3     String url = "http://localhost:8080/fhir/Observation";
4
5     CloseableHttpClient httpClient = HttpClients.createDefault();
6     HttpPost httpPost = new HttpPost(url);
7
8     JSONArray ecg = new JSONArray();
9     for(double e: bsd.getEcg()){
10        ecg.put(new JSONObject().put("high", new JSONObject().put("
value", e)));
11    }
12
13    JSONArray component = new JSONArray();
14    JSONObject ecgFrequency = new JSONObject();
15    ecgFrequency.put("code", new JSONObject().put("text", "
ecgFrequency"));
16    ecgFrequency.put("valueQuantity", new JSONObject().put("value",
bsd.getEcgFrequency()));
17    (...)
18    component.put(ecgFrequency);
19    component.put(heartRate);
20    component.put(heartRateVariability);
21    component.put(respirationRate);
22
23    SimpleDateFormat sm = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss"
);
24
25    String body = new JSONObject()
26        .put("resourceType", "Observation")
27        .put("subject", new JSONObject().put("reference", "Patient/"
+ bsd.getParticipantId()))
28        .put("effectivePeriod", new JSONObject().put("start", sm.
format(bsd.getStartDate())).put("end", sm.format(bsd.getEndDate())))
29        .put("valueInteger", bsd.getStance())
30        .put("referenceRange", ecg)
31        .put("component", component)
32        .put("code", new JSONObject().put("text", "BioSensor"))
33        .toString();
34
35    StringEntity params = new StringEntity(body);
36    params.setContentType("application/json");
37
38    httpPost.addHeader("content-type", "application/json");
39    httpPost.addHeader("accept", "application/json");
40    httpPost.setEntity(params);
41
42    CloseableHttpResponse response = httpClient.execute(httpPost);
43
44    (...)
45
46    return res;
47
48 }
49

```

Listing 5.14: Data Service BiosensorData Persistence

Following this use case, we have **UC8**, which also has a complex body setup, as the Sleep+ and Bedroom+ Data will also comprise of a FHIR Observation.

```

1 public String createSleepBedroomData(SleepBedroomData sbd) throws
    ClientProtocolException, IOException, ParseException {
2
3     String url = "http://localhost:8080/fhir/Observation";
4
5     CloseableHttpClient httpClient = HttpClients.createDefault();
6     HttpPost httpPost = new HttpPost(url);
7
8     JSONArray component = new JSONArray();
9     JSONObject sta = new JSONObject();
10    sta.put("code", new JSONObject().put("text", "sta"));
11    sta.put("valueQuantity", new JSONObject().put("value", sbd.
        getSleepPlus().getTemperature().getAverage()));
12    component.put(sta);
13    JSONObject stmin = new JSONObject();
14    stmin.put("code", new JSONObject().put("text", "stmin"));
15    stmin.put("valueQuantity", new JSONObject().put("value", sbd.
        getSleepPlus().getTemperature().getMin()));
16    component.put(stmin);
17    JSONObject sleepTempMax = new JSONObject();
18    stmax.put("code", new JSONObject().put("text", "stmax"));
19    stmax.put("valueQuantity", new JSONObject().put("value", sbd.
        getSleepPlus().getTemperature().getMax()));
20    component.put(stmax);
21    (...)
22
23    SimpleDateFormat sm = new SimpleDateFormat("yyyy-MM-dd'T'HH:mm:ss");
24
25    String body = new JSONObject()
26        .put("resourceType", "Observation")
27        .put("subject", new JSONObject().put("reference", "Patient/" +
        sbd.getUserId()))
28        .put("effectivePeriod", new JSONObject().put("start", sm.format(
        sbd.getStartReading()).put("end", sm.format(sbd.getEndReading()))
29        .put("issued", sm.format(sbd.getSent()))
30        .put("component", component)
31        .put("code", new JSONObject().put("text", "SleepBedroom"))
32        .toString();
33
34    StringEntity params = new StringEntity(body);
35    params.setContentType("application/json");
36    (...)

```

Listing 5.15: Data Service Sleep+ and Bedroom+ Data Persistence

As can be viewed in Listing 5.15, the majority of the request setup for the persistence of Sleep+ and Bedroom+ Data lies upon the various temperature, humidity and head position values. For each of these, a JSON Object will be created, containing a code identifying what the measurement is, and a value, with said JSON Object being inserted in a JSON Array (examples for the Sleep+ Temperature Maximum, Minimum and Average values in lines 8 to 20, with the remaining measurements following the same pattern but being omitted in the listing for brevity). Once the JSON Array is filled with the measurements, it is added to the main body of the request (line 30), along with other relevant data, such as the patient the

measurements belong to (line 27), the start and end of the measurement period (line 28), the date the measurements were issued (line 29) and a code identifying the Observation as Sleep+ and Bedroom+ Data (line 31). Afterwards, the request follows suit as the previous examples did.

As for **UC6**, **UC7** and **UC9**, all these relate to finding information by a criteria. The first one, consists in finding all the patients that are associated to a specific clinical trial, while the remaining two are about finding all the Biosensor Data and Sleep+/Bedroom+ Data measurements associated to a specific patient. The methods required for each of these data retrieval use cases follow the same structure, evidenced in Listing 5.16, which features the method for getting all the patients of a clinical trial.

```
1 public String getPatientsByTrial(String org) throws
2   ClientProtocolException, IOException, ParseException {
3     String url = "http://localhost:8080/fhir/Patient?organization="+ org
4     ;
5     CloseableHttpClient httpClient = HttpClients.createDefault();
6     HttpGet httpGetCount = new HttpGet(url);
7
8     httpGetCount.addHeader("content-type", "application/json");
9     httpGetCount.addHeader("accept", "application/json");
10
11    CloseableHttpResponse requestResponse = httpClient.execute(
12    httpGetCount);
13
14    HttpEntity ent = requestResponse.getEntity();
15
16    String res = EntityUtils.toString(ent, "UTF-8");
17
18    JSONObject resultJSON = new JSONObject(res);
19
20    int total = Integer.parseInt(resultJSON.get("total").toString());
21
22    url = "http://localhost:8080/fhir/Patient?organization="+ org + "&
23    _count=" + total;
24
25    HttpGet httpGet = new HttpGet(url);
26
27    (...)
28
29    return res;
30 }
```

Listing 5.16: Data Service Getting Patients by Clinical Trial

The first aspect to consider is that the FHIR Server uses pagination when responding to queries, which means that upon receiving requests for data, it will by default only respond in parts of 20 entries (if, for example, there are 60 patients in the system and the request asks for all the patients, the response will have the first 20 patients, unless additional parameter criteria are added to the URL of the request, to further the response pool). As such, in order to obtain the full set of results to a query, the first thing to do is discover the total amount of possible results (which is what is being done in lines 3 to 19) by requesting the

FHIR Server for all the Patients with an association to an Organization (the FHIR Resource Type for a Clinical Trial) with the ID of the Clinical Trial in question, with the response being extracted from the request and the "total" value in it being isolated.

Knowing now the total amount of patients that qualify for our query (how many Patients are associated to that particular Clinical Trial), this value can now be used in the parameter of the URL of the request, as the "_count" criteria (line 21), which will let us this time obtain the full response set.

Lastly, we have **UC5**, where the persistence of a Bundle of Clinical Data must be addressed. For this use case, the Data Service expects to receive an array of JSON Objects, each pertaining to a Resource Type from FHIR and a type of transaction (GET, POST, PUT or DELETE) that must be executed with that Resource Type data. Essentially, this method is supposed to serve as a more direct, though complex way to relay large amounts of data to the FHIR Server, bypassing the restructuring of data that the API offers and simply providing direct FHIR-ready request bodies. For example, the method could be used to persist the data of dozens of patients of a certain trial, followed by the creation of a new trial, by submitting something like the body in Listing 5.17.

```
1 [
2   {
3     "resource":{
4       "resourceType":"Patient",
5       "managingOrganization":{"reference":"189"}
6     },
7     "request":{
8       "method":"POST",
9       "url":"Patient"
10    }
11  },
12  {
13    "resource":{
14      "resourceType":"Patient",
15      "managingOrganization":{"reference":"189"}
16    },
17    "request":{
18      "method":"POST",
19      "url":"Patient"
20    }
21  },
22  (...)
23  {
24    "resource":{
25      "resourceType":"Organization"
26    },
27    "request":{
28      "method":"POST",
29      "url":"Organization"
30    }
31  }
32  (...)
33 ]
```

Listing 5.17: Example of the body required by the Bundle method in the Data Service

By sending this bundle to the FHIR Server, it will be able to sequentially create the Patients and then the Trial in a single operation of the API. This is, by recognizing their resource types, validating their properties accordingly and then using the request details ("method" and "url") to locate the proper resource provider and its coded method capable of complying with the desired action.

The method that handles this expects to receive a body such as the one presented in the previous listing, setting up the proper HTTP request to encase it. In Listing 5.18, we can see such a method, where the URL to be used is the general FHIR Server path (line 3), as no specific Resource Provider will be targeted, and the body of the request will identify its contents as a Bundle (line 9) and insert the provided array of resources/requests as the entry value of the bundle (line 11).

From that point on, the method functions as expected, executing the request and extracting, logging and returning the response to it, which includes the individual response to each request described in the bundle entry.

```
1 public String postBundle(String str) throws ClientProtocolException ,
   IOException , ParseException {
2
3     String url = "http://localhost:8080/fhir";
4
5     CloseableHttpClient httpClient = HttpClients.createDefault();
6
7     HttpPost httpPost = new HttpPost(url);
8
9     String body = "{\"resourceType\" : \"Bundle\", \" +
10    \"type\" : \"transaction\", \" +
11    \"entry\" : \" + str + \"}\"";
12
13    StringEntity params = new StringEntity(body);
14
15    params.setContentType("application/json");
16
17    httpPost.addHeader("content-type", "application/json");
18
19    httpPost.addHeader("accept", "application/json");
20
21    httpPost.setEntity(params);
22
23    CloseableHttpResponse response = httpClient.execute(httpPost);
24
25    String res = EntityUtils.toString(response.getEntity(), "UTF-8")
26    ;
27
28    logger.info(res);
29
30    return res;
31 }
```

Listing 5.18: Data Service Approach to Posting Bundles

5.3.3 Controller

The Data Controller is responsible for receiving and interpreting the various requests sent to the API. Its methods all follow the same structure (with only some slight differences). Each one has a mapping value, to set the connection between the URL path and the method itself, and each will return a **Response Entity** object, which will either flag the request as successful, in which case it returns a response containing the requested/provided data, or flag the request as unsuccessful, indicating the exception that caused it and its details. Also, depending on the method, some might receive information to be used by the Data Service, either as a **Path Variable** or as a **Parameter** in the **Request Body**.

To serve as example to this, we have an extract of the Data Controller in Listing 5.19.

```
1 @RestController
2 @RequestMapping("/api")
3 public class DataController {
4
5     @Autowired
6     DataService dataService;
7
8     @ApiOperation(value = "Create Trial")
9     @PostMapping("/trial")
10    public ResponseEntity createEnsaio () {
11        try {
12            String s = dataService.createEnsaio ();
13            return new ResponseEntity <>(s, HttpStatus.CREATED);
14        } catch (Exception e) {
15            return new ResponseEntity <>(HttpStatus.ALREADY_REPORTED);
16        }
17    }
18
19    @ApiOperation(value = "Persist Biosensor Data")
20    @PostMapping("/biosensordata")
21    public ResponseEntity createBiosensorData (@RequestBody BioSensorData
22        bsd) {
23        try {
24            String s = dataService.createBiosensorData(bsd);
25            return new ResponseEntity <>(s, HttpStatus.CREATED);
26        } catch (Exception e) {
27            return new ResponseEntity <>(HttpStatus.ALREADY_REPORTED);
28        }
29    }
30
31    @ApiOperation(value = "Obtain patients associated to specific trial")
32    @GetMapping("/patients/trial/{org}")
33    public ResponseEntity getPacientes (@PathVariable("org")
34        @ApiParam(value = "Trial ID", example = "000") String id) {
35        try {
36            String s = dataService.getPatientsByTrial(id);
37            return new ResponseEntity <>(s, HttpStatus.OK);
38        } catch (Exception e) {
39            return new ResponseEntity <>(HttpStatus.ALREADY_REPORTED);
40        }
41    }
42 }
```

Listing 5.19: Data Controller Extract

In this listing, we start by flagging the controller as a **Rest Controller**, so that the Spring Boot Framework recognizes it and uses it as such (line 1), and give it the **Request Mapping** of `"/api"`, so that any requests directed to this controller must have their URL start with this sequence after the port number (line 2). After this, we have the Data Service that the controller will use, which is **Autowired** (lines 5 and 6) and a basic constructor (lines 9 to 11).

With all the essential setup, we then have the three given examples of controller methods. The first one is for the creation of Clinical Trials. It is given a title with **ApiOperation** (line 13), to be used in the Swagger of the API, and a mapping, in particular a **PostMapping**, of `"/ensaio"` (line 14), so that any POST requests made to `"api/ensaio"` will be redirected to this method. The method itself is covered by a try/catch sequence, where the method will try to invoke the creation of a trial from the service (line 18), returning the details of the new trial and a **201 Created** status if successful (line 20), or simply a **208 Already Reported** status if not successful (line 22).

The second example follows the same logic, with the difference being that it receives a parameter object, flagged by the **RequestBody** annotation, so that the system knows to extract the data from the JSON body of the request and use it to create a `BioSensorData` object.

The third example also follows the same logic, with some key differences. It also has a title through `ApiOperation`, but its mapping is a **GetMapping** (line 27), seeing how the method is intended for GET requests. The mapping will also contain a section in brackets (`"org"`) which is identified as a **PathVariable** (line 28), meaning that if the URL is `"api/patients/-trial/123"`, the value `"123"` will be saved in the string variable `"id"` (line 29). An **ApiParam** is also used to codify a name for this parameter in Swagger and an example value to be displayed in that page. The last key difference is the HTTP status that is returned in a successful situation, which this time around is **200 OK**.

5.3.4 Swagger Setup

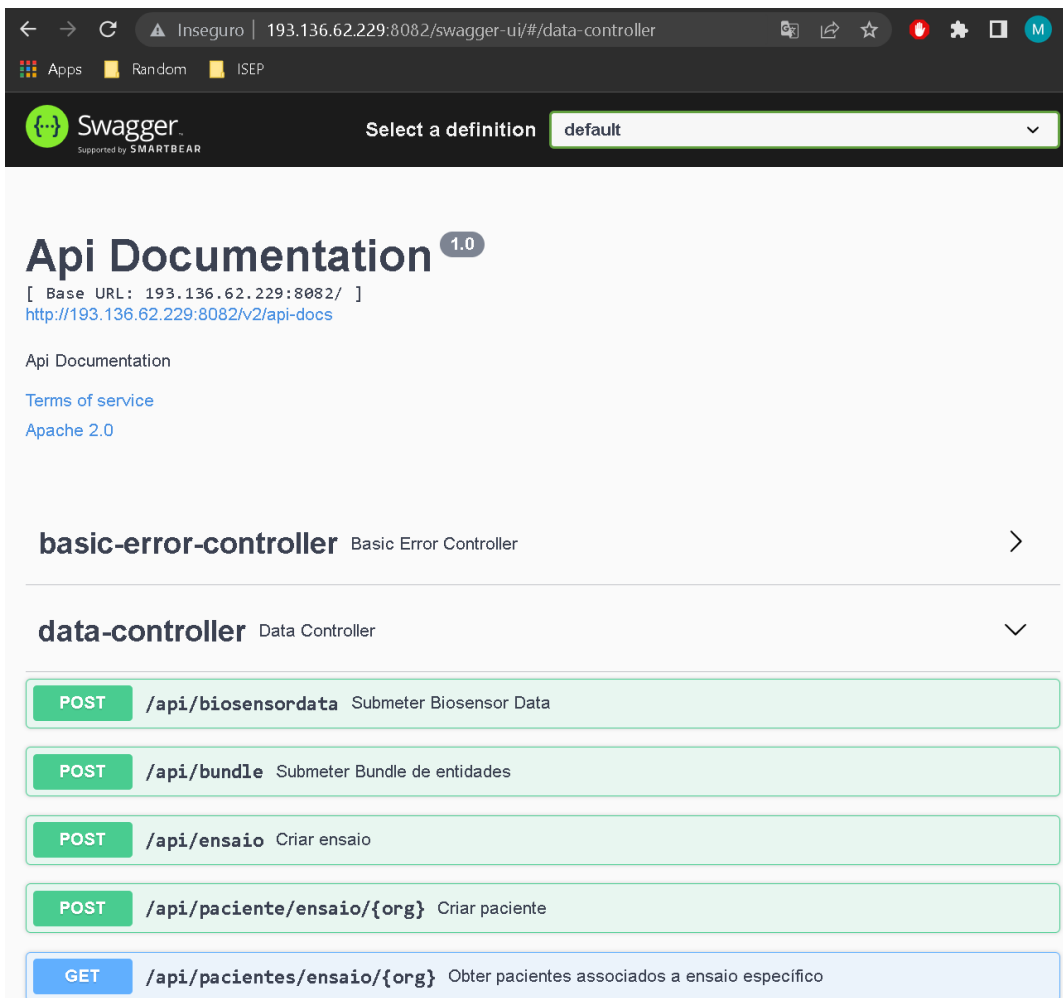
As the platform is mainly intended to be used directly by the applications, services and devices of Smart-Health technology, there was no crucial need for the development of a UI. However, for cases where data is inserted manually by a user, such as in situations of maintenance, as well as for documentation purposes, a Swagger page was considered. This page would contain descriptions of the various endpoints of the Maintenance API and their respective details, such as the type of request and their requirements, along with options for execution in the page itself.

As described in the previous subsection, the Data Controller code contain annotations that set the various methods of the controller as API Operations recognized by Swagger, identifying their names, with other annotations also identifying relevant aspects to Swagger, such as parameter names and example values. However, for the API to actually recognize these Swagger annotations and render a Swagger page, a configuration class, such as the one in Listing 5.20, must be produced.

```
1 @Configuration
2 @EnableSwagger2
3 public class SwaggerConfig {
4     @Bean
5     public Docket api() {
6         return new Docket(DocumentationType.SWAGGER_2)
7             .select()
8             .apis(RequestHandlerSelectors.any())
9             .paths(PathSelectors.any())
10            .build();
11    }
12 }
```

Listing 5.20: Swagger Configuration Class

The Swagger page is accessible by adding `"/swagger-ui/#/"` to the URL of the management API, following the port number, as can be viewed in Figure 5.2. For each endpoint, the data is presented according to what was coded in the respective controller method, as shown in Figure 5.3.



The screenshot displays the Swagger UI interface for the Management API. The browser's address bar indicates the URL `193.136.62.229:8082/swagger-ui/#/data-controller`. The page features a dark header with the Swagger logo and a dropdown menu for selecting a definition, currently set to 'default'. The main content area is titled 'Api Documentation 1.0' and provides the base URL `http://193.136.62.229:8082/v2/api-docs`. Below the header, there are two sections: 'basic-error-controller' (Basic Error Controller) and 'data-controller' (Data Controller). The 'data-controller' section is expanded to show a list of endpoints:

- POST** `/api/biosensordata`: Submeter Biosensor Data
- POST** `/api/bundle`: Submeter Bundle de entidades
- POST** `/api/ensaio`: Criar ensaio
- POST** `/api/paciente/ensaio/{org}`: Criar paciente
- GET** `/api/pacientes/ensaio/{org}`: Obter pacientes associados a ensaio específico

Figure 5.2: Management API Swagger Page

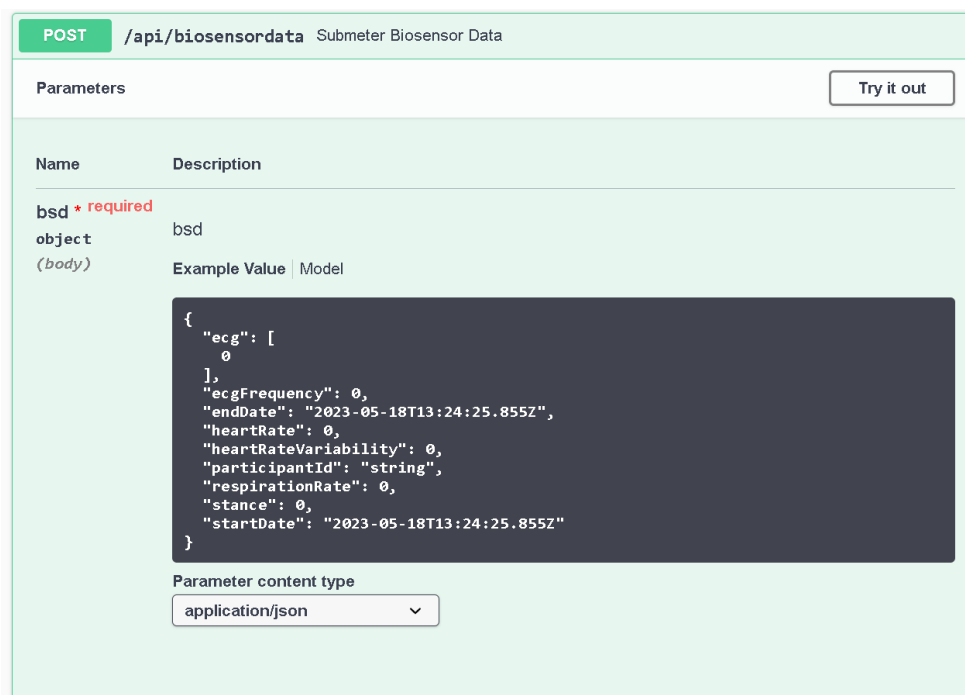


Figure 5.3: Management API Swagger Page Endpoint

5.4 Running the Platform

The final section of the implementation will explain how to execute the platform, looking specifically at how to run the HAPI FHIR Server and the Management API.

It is important to note that the platform will be operating in a Linux Virtual Machine which is accessed remotely. As such, the presented commands take into consideration the need for the components of the platform to run even if the session is interrupted/disconnected.

5.4.1 Running the Server

In order to run the server locally, as presented in Listing 5.21, the terminal must be set in the directory of the server (line 1), and the run command of Spring Boot must be launched (line 2).

```
1 $ cd SH4ALL_HL7_Prototype_JPAServer
2 $ sudo mvn clean spring-boot:run
```

Listing 5.21: Commands to locally run the HAPI FHIR Server

If a Virtual Machine is being used and accessed remotely, the command must be adjusted, to include the "nohup" prefix, standing for "no hangup", which prevents the command from being aborted automatically when the session is exited. The adjusted command can be seen in Listing 5.23.

```
3 $ cd SH4ALL_HL7_Prototype_JPAServer
4 $ sudo nohup mvn clean spring-boot:run -Pboot
```

Listing 5.22: Commands to remotely run the HAPI FHIR Server

5.4.2 Running the Management API

To run the Management API, the same logic as the server applies. To run locally, as seen in Listing 5.22, access the directory where the API was developed and simply use the run command. To run remotely, as in Listing 5.24, access the directory and use the run command alongside the "nohup" prefix.

```
5 $ cd api
6 $ sudo mvn clean spring-boot:run
```

Listing 5.23: Commands to locally run the Management API

```
7 $ cd api
8 $ sudo nohup mvn clean spring-boot:run -Pboot
```

Listing 5.24: Commands to remotely run the Management API

Chapter 6

Experimentation and Evaluation

This chapter will go over the process of reviewing, testing and evaluating the implemented solution. Specifically, indicators, which are aspects/segments of the solution, were studied in order to confirm hypothesis, which are potential outcomes of the implementation, using certain methods of evaluation. From this, conclusions were made about the solution's capacity to meet a set of expectations.

6.1 Purpose

Any software platform, upon being implemented, needs to be tested, not only to verify the proper execution of its functions, but also other parameters, such as the efficiency in performance.

This procedure will be applied to the data integration platform, implemented in the previous chapter. Since the entire purpose of the platform is to facilitate and fast-track the assimilation of clinical data, which would previously have to be done in a less automated and efficient way, evaluating it becomes an even more relevant step, due to expectations of its performance level that were established early on in its conceptualization.

6.2 Indicators

Indicators are used to segment the platform in testing into the aspects which need to be reviewed and evaluated. Specifically, these should be characteristics, behaviours or functions of the app which can be studied, measured or qualified in a series of assessments, according to an evaluation methodology.

The indicators chosen for the evaluation of the platform were:

- **The proper storage of clinical data, its relations and references (Functionality):** If the platform is indeed functioning as it is supposed to, by dissecting the server's data after numerous data inputs, the data should be displayed according to the FHIR resource architecture, while retaining expected entity relationships and references.
- **Performance with consecutive small to medium requests:** The platform should have the capability of processing data efficiently, despite different levels of size and complexity, ranging from small inputs, such as a patient data, to larger ones, such as Biosensor data. The system should be prepared to handle these requests in succession, as real-life use of the system will very likely entail a high-rate of inputs in short intervals, without losing performance efficiency. This means that all these consecutive inputs

should be processed in less than 1 second each, no matter what type of data is being persisted.

- Performance with large bundles of entities and resources:** Considering the capacity of the platform to persist bundles, the performance should still be optimal, no matter the bundle that is being persisted. Taking to account that a bundle has no real limit to its size and complexity, we can have JSON files for bundles that are only half a dozen lines in length, or files that are thousands upon thousands of lines in length. Either way, even with increasing degrees of data, both in size and number, the platform should still reach an expected level of speed and performance, never taking more than 10 minutes to fully process a bundle file.

6.3 Hypothesis

Having defined the indicators for the experimentation, a set of hypothesis will be defined for each. An hypothesis will represent a possible outcome of the experimentation, which will be proven, or not, in its execution.

For each indicator, two opposing hypothesis were determined. One of these is the null hypothesis, represented by H0, while the other is the alternative hypothesis, represented by H1. These are presented in Table 6.1:

Table 6.1: Hypothesis for Experimentation and Evaluation

Indicator	H0	H1
Proper storage of clinical data	Data is stored improperly/incomplete or non-existent	Data is completely stored and properly assigned to the correct FHIR resources
Performance (consecutive small to medium requests)	Platform demonstrates inefficient performance, with speed of storage falling below expected parameters (taking more than 200 milliseconds for sequential requests)	Platform proves to be efficient, storing data with speed equal or above expected values, even with varying data size and complexity (less than 200 milliseconds per request)
Performance (large bundles)	Platform demonstrates inefficient performance, with speed of storage falling below expected parameters (taking more than 10 minutes to persist a bundle)	Platform proves to be efficient, storing data with speed equal or above expected values, even with varying data size and complexity (bundle persistence exceeding 10 minutes)

6.4 Evaluation Methodology

To understand the experiments that will be executed to confirm the listed hypothesis, the methodology will be explained, indicating which experiments will be made, with which data sets and tools and how the results will be interpreted.

6.4.1 Process

The indicators listed in Chapter 6.2 will be experimented on in different ways.

To test the proper functionality of the system (the first indicator), **System Tests and Acceptance Tests** will be executed. For these tests, the platform will be provided with a wide array of inputs, starting with simple ones and consequently growing in complexity, verifying that the inputs are not only processed and the ideal response is given (System Test) but the data is properly stored in the FHIR Server according to its intended resource and relationships with other resources (Acceptance Test).

For the testing of both of the Performance indicators, the system will be provided with multiple data-sets to analyse its resilience and efficiency in processing them.

For the performance with consecutive small to medium requests, this will include its capacity to process a large number of small to medium consecutive inputs, namely 1000 requests, which will include creation of trials and patients and the persistence of Biosensor and Sleep+/Bedroom+ data sets. It should be able to receive and process these requests, without shutting down or messing up the data, and taking no longer than 20 milliseconds for each.

The performance with large bundles will be tested by submitting multiple considerably sizable bundles (namely, 30 bundles, with the largest bundle's JSON file being one with 1480376 lines and 43 Megabytes in size). Faced with these large, complex inputs, the platform should be capable of maintaining its processing, taking no more than 8 minutes for each.

As such, the **Performance Tests** will have the timing of the platform under these two circumstances, with the timing values being subject to statistical evaluation and review afterwards, in an effort to confirm or deny their hypothesis.

6.4.2 Tools

To execute the input of information required in the described tests, *Postman* (2023) will be used. This application comes with a lot of perks that will aid the tests and the capture of data, such as the ability to prepare, orchestrate and schedule requests to the platform, view response data in detail and time the execution times of said requests.

For the statistical analysis of the captured data, the experiments will make use Microsoft Excel, not only to store and organize said data, but perform calculations and provide the necessary statistical results.

To display the statistical results in visually accessible ways, the use of *R* (2023), a statistical computing and graphics software, will be essential, using it to create graphs and other documentation.

6.4.3 Interpretation of Results

For the Functionality indicator, the results from tests will be presented in statistics (tables and graphs) showcasing the success rate of these, in regard to growing complexity of inputs. The H1 hypothesis will be confirmed if the results are favorable (with a full success rate), otherwise H0 will be the verified hypothesis.

For the Performance indicators, statistics will be obtained using the timing of the platform operations. Evaluating measures, such as means, medians and averages, as well as various tests, will be used to determine the normality and the location of these values, leading to the confirmation of either hypothesis H0 or H1.

6.5 Experiments and Result Analysis

This section will detail the execution of the tests that were previously described and the statistical review of their results.

6.5.1 System Tests and Acceptance Tests

The System Tests required the simulation of real-life input and retrieval of data with the platform. Postman was used to simulate the Smart-Health technology that would in normal conditions be part of the interaction. Postman would have a sequence of requests that would be executed in the given order and would attempt to validate the responses given by the platform, so that when provided with correct or incorrect information, the platform would respond as expected.

In Figure 6.1, the setup of a singular test can be seen, where the required data has been provided (URL, body and headers) and the test configuration sets the values to be checked (HTTP status) and the expected values, passing the test if these are equal.



Figure 6.1: Example of Postman System Test

In Table 6.2, the details of the full set of System Tests that were developed with Postman can be seen (including the expected HTTP status code and, in certain cases, the expected response body). since these are executed sequentially, it is expected that an earlier test might influence a later one. For example, the test involving the creation of a Patient will impact a later test that persists Biosensor Data to that same Patient.

Postman interprets each expectation described in the Tests as a singular test. As such, the Test described in Test5 actually represents two tests, one for the status value and another

Table 6.2: System Tests Summary Table

Test	Description	Expected Status and Response
1	POST of a Clinical Trial A	201
2	POST of a Patient A (associated to trial A)	201
3	POST of a Clinical Trial B	201
4	POST of a Patient B (associated to trial A)	201
5	POST of an Invalid Patient	208 (Response containing Error due to association to non-existent trial)
6	GET of Patients in Trial A	200 (Response containing list with Patient created in Test 2 and 4)
7	GET of Patients in Trial B	200 (Response containing empty list)
8	POST of Biosensor Data to Patient A	201
9	POST of Invalid Biosensor Data	208 (Response containing Error due to association to non-existent patient)
10	GET of Biosensor Data of Patient A	200 (Response containing list with data created in Test 8)
11	GET of Biosensor Data of Patient B	200 (Response containing empty list)
12	POST of SleepBedroom Data to Patient A	201
13	POST of Invalid SleepBedroom Data	208 (Response containing Error due to association to non-existent patient)
14	GET of SleepBedroom Data of Patient A	200 (Response containing list with data created in Test 12)
15	GET of SleepBedroom Data of Patient B	200 (Response containing empty list)

for the contents of the list returned in the response body. Knowing this, Postman ran a total of **24 tests** of which **24 passed**, leading to a **100% success rate** in the Systems tests.

This leads us to the Acceptance tests, which made use of the requests already executed in the System tests and consisted in manually accessing the FHIR server to verify the proper storage of the Trials, Patients and Biosensor and SleepBedroom data sets that were persisted.

In fact, the Acceptance Tests also had a **success rate of 100%**, with all the data having been accurately processed and stored. An example of the manual verification that was done can be seen in Figure 6.2, where the Biosensor Data that was created being displayed by the FHIR Server (as a FHIR Observation) and its relation to Patient A being also present.



```

{
  "resourceType": "Observation",
  "id": "102",
  "meta": {
    "versionId": "1",
    "lastUpdated": "2023-05-22T08:37:06.511+00:00",
    "source": "#vLPsu25Qd4gqzpdW"
  },
  "code": {
    "text": "SleepBedroom"
  },
  "subject": {
    "reference": "Patient/3"
  },
  "effectivePeriod": {
    "start": "2021-01-01T11:11:11",
    "end": "2021-01-01T11:11:11"
  },
  "issued": "2021-01-01T11:11:11",
  "component": [ {
    "code": {
      "text": "sleepTempAvg"
    },
    "valueQuantity": {
      "value": 0
    }
  }, {
    "code": {
      "text": "sleepTempMin"
    },
    "valueQuantity": {
      "value": 0
    }
  }, {
    "code": {
      "text": "sleepTempMax"
    },
    "valueQuantity": {
      "value": 0
    }
  }
]
}

```

Figure 6.2: Example of Manual Verification for the Acceptance Test

Taking into account the success of the System and Acceptance tests, we can address the hypothesis for the Proper Storage of Clinical Data indicator, listed in Table 6.1, and reject H0, while accepting H1.

6.5.2 Performance Tests for Consecutive Requests

These performance tests will be based around the duration of each request executed, in milliseconds, therefore dealing with a quantitative data type, and a ratio type of measurement.

In total, 1000 individual requests were made in succession. All of these tests represented small to medium operations, ranging from the creation of clinical trials and patients, to the persistence of measurements such as Biosensor and SleepBedroom Data. The pool of requests that were executed were defined with the help of the stakeholders of the project, in accordance with realistic circumstances, so that these could simulate the data input in regular, real-life conditions.

The full set of requests, their descriptions and respective duration can be found in full capacity in the Appendix B. By reviewing these values, some statistical analysis can already be done. The summary of these initial statistical conclusions is shown in Table 6.3, including the total size of the sample, as well as the **Mean**, **Median** and **Standard Deviation** values that were obtained.

Table 6.3: Summary of the data of the Performance Tests for Consecutive Requests

Data Parameter	Value (ms)
Sample Size	1000
Minimum	40
Maximum	210
Mean	123.809
Median	124
Standard Deviation	48.44983

A visual representation of the findings of Table 6.3 is also possible, by using a box plot graph, as in Figure 6.3. It gives us other pieces of data, such as the Lower Quartile (Q1) and Upper Quartile (Q3), which were 82 ms and 164 ms, respectively.

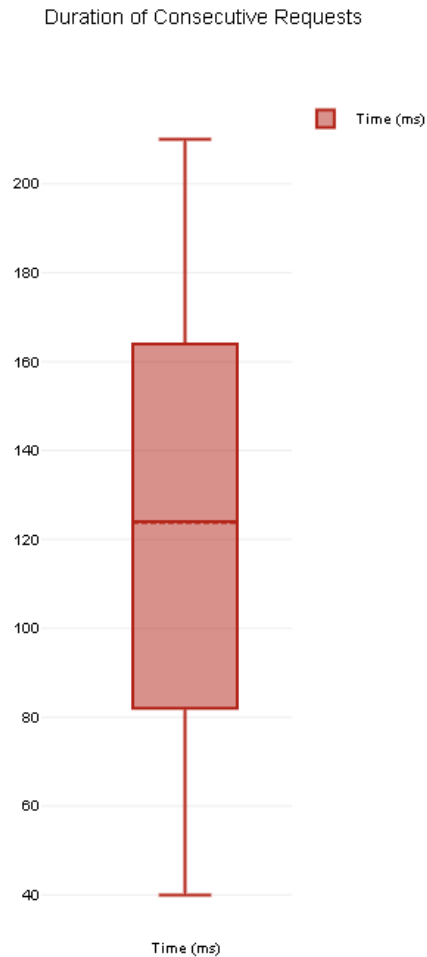


Figure 6.3: Boxplot Graph for data obtained from Performance Tests for Consecutive Requests

Now that these statistical measures have been taken, the initial hypothesis relating to the performance tests for consecutive requests must be addressed. To do this, first the normality of the distribution of the sample must be checked. As such, for this verification, we have two hypothesis: **H0**, which dictates that "there is normal distribution", and **H1**, where "normal distribution does not apply". To verify and reject one of these, two tests were considered, the **Shapiro-Wilk** test and the **Kolmogorov–Smirnov** test. The prior is more applicable to smaller samples, while the prior is preferable for larger ones. As we are dealing with a sample of 1000 entries, the Kolmogorov–Smirnov test was applied (formula is depicted in equation 6.1).

$$D = \max_{1 \leq i \leq N} \left(F(Y_i) - \frac{i-1}{N}, \frac{i}{N} - F(Y_i) \right) \quad (6.1)$$

By executing the test with the given sample data, we obtain a value of **D = 0.06236** and a **p-value = 0.0008017**. Considering that the **Significance level (α)** is of 0.05, we can claim that $p\text{-value} < \alpha$, therefore rejecting the H_0 and determining the data distribution as **not normal**.

Knowing now about the distribution, we can select how to test the hypothesis listed in Table 6.1, with **H0** being "Platform demonstrates inefficient performance, with speed of storage falling below expected parameters (taking more than 200 milliseconds for sequential requests)" and **H1** being "Platform proves to be efficient storing data with speed equal or above expected values even with varying data size and complexity (less than 200 milliseconds per request)". With the non-normal distribution, we will a non-parametric test, namely the **Wilcoxon** test.

This test will take into consideration the provided sample and the intent of verifying the tendency of the values of the sample to exceed 200 milliseconds. The R command used to achieve this is present in Listing 6.2:

```
1 data <- c(...)  
2  
3 res <- wilcox.test(data, mu = 200, alternative = "less")  
4  
5 res
```

Listing 6.1: Wilcoxon Test performed in R

The results tell us that **p-value = 2.2e-16** which means that the p-value is lesser than the significance level (0.05), allowing us to reject H_0 and support H_1 .

As such, the Performance Tests for Consecutive Requests can be seen as successful, with the ideal hypothesis being proven.

6.5.3 Performance Tests for Bundles

For this set of Performance Tests, Bundles were prepared, with the specifications given by stakeholders, to verify the resilience and efficiency of the platform in processing them. Each bundle aimed to simulate a "data dump", containing multiple sets of information about clinical trials, patients, data measurements and other resources the FHIR Server might recognize.

A total of 30 tests were executed, having 30 bundles been set up, the smallest of which was 10798 lines in length and 309 Kilobytes in size, and the largest had 1480376 lines and 43 Megabytes.

The processing time of the request that sent the bundle to the platform is going to be the variable considered here. Table 6.4 details the bundle of each test, indicating its size, line count and the time it took for its complete processing, in minutes and seconds, as well as in total amount of seconds (the total in seconds will be used for later calculations). In the table, there were also included some of the conclusions taken from the results of the requests, such as the limits, mean and median of the durations.

Table 6.4: Summary of the data of the Performance Tests for Bundles

Test/Bundle	Size	Lines	Time (MM:SS)	Time (Seconds)
1	309 KB	10798	00:15	15
2	432 KB	11561	00:17	17
3	540 KB	13002	00:22	22
4	675 KB	14599	00:23	23
5	733 KB	17601	00:26	26
6	820 KB	19101	00:29	29
7	992 KB	20918	00:41	41
8	1 MB	22012	00:49	49
9	3 MB	71100	01:06	66
10	4 MB	92116	01:19	79
11	7 MB	218232	01:37	97
12	9 MB	291084	02:00	120
13	10 MB	310155	02:24	144
14	12 MB	443026	02:42	162
15	15 MB	537075	03:00	180
16	18 MB	723359	03:17	197
17	19 MB	769172	03:43	223
18	21 MB	927730	04:05	245
19	22 MB	948711	04:26	266
20	24 MB	1016674	04:58	298
21	26 MB	1079981	05:16	316
22	28 MB	1124098	05:26	326
23	30 MB	1198691	05:38	338
24	32 MB	1222854	05:59	359
25	35 MB	1267832	06:11	371
26	36 MB	1308911	06:35	395
27	38 MB	1340292	06:52	412
28	40 MB	1396722	07:09	429
29	41 MB	1399374	07:15	435
30	43 MB	1480376	07:30	450
Sample Size			30	
Minimum			15	
Maximum			450	
Mean			204.3(3)	
Median			188.5	
Standard Deviation			148.87	

Similarly to the previous set of performance tests, the results were also used to develop a box-plot graph, showcasing the Minimum, Maximum, Mean and Median values, alongside the Lower and Upper Quartile, which were 49 and 338. The graph can be viewed in Figure 6.4.

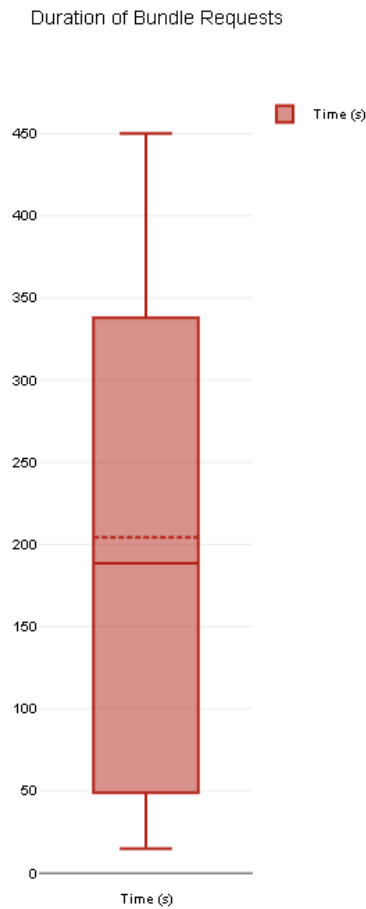


Figure 6.4: Boxplot Graph for data obtained from Performance Tests for Bundles

The distribution of the results must now be verified, following the logic of the following hypothesis: **H0**, which dictates that "there is normal distribution", and **H1**, where "normal distribution does not apply". Considering that the sample size is 30, between the options of the Shapiro-Wilk test and the Kolmogorov-Smirnov test, the prior was chosen due to being preferable for smaller samples ($n < 50$).

The formula for the Shapiro-Wilk test can be seen in Equation 6.2, using the observations of the sample (y), their total number (n) and tabulated coefficients (a) :

$$W = \frac{(\sum_{t=2}^n a_t y_t)^2}{\sum_{t=1}^n (x_t - \bar{y})^2} \quad (6.2)$$

By executing the Shapiro-Wilk test with the values given in Table 6.4, we obtain the values of **W = 0.8974** and **p-value = 0.00724434**. Since $p\text{-value} < 0.05$, it is lower than the significance level, which leads to the rejection of H_0 and the confirmation that the data set we are working with is **not normal** in its distribution.

This takes us to the execution of **Wilcoxon Test** upon the data set, considering the hypothesis previously presented for these Performance Tests: **H0** as "Platform demonstrates inefficient performance, with speed of storage falling below expected parameters (taking more than 10 minutes persist a bundle)" and **H1** as "Platform proves to be efficient, storing data with speed equal or above expected values, to even with varying data size and complexity (bundle persistence exceeding 10 minutes)".

The Wilcoxon test will use the default value mentioned in the hypothesis, which was 10 minutes, equalling to 600 seconds. In Listing 6.2, the commands used in R to run and review the results of the test can be seen.

```
1 data2 <- c(...)  
2  
3 res2 <- wilcox.test(data2, mu = 600, alternative = "less")  
4  
5 res2
```

Listing 6.2: Wilcoxon Test performed in R

Having run the commands, we are told **p-value = 9.313e-10**, meaning $p\text{-value} < \alpha$. Knowing this, we can reject H_0 and support H_1 .

As such, we can stand by the fact that the platform manages to handle its bundles with the required level of efficiency.

6.6 Final Evaluation

Having finalized the three sets of tests that were initially detailed, the following conclusions can be had:

- Regarding the indicator of **proper storage of clinical data**, the system tests and acceptance tests showed that the platform was fully capable of executing the complete business logic and properly handle all the persistence of data and its consequent retrieval. The responses to each request were reviewed and validated, as was the integrity of the database following these same requests, proving that the data was stored as idealized;
- When reviewing the first **performance** indicator, which looked at the resilience and response time of the platform when dealing with sequential requests, it was determined that it was capable of handling a vast amount of them in quick succession without faltering in its response time. By annotating these duration values and statistically reviewing them, the level of performance met the expectations that were previously set;

- As for the second, and last **performance** indicator, which focused on the handling of bundles by the platform, verifying its capacity in processing large chunks of data, the annotated time intervals for the platform's reception and persistence of the bundles was also reviewed and statistically explored, to confirm the expected performance levels.

All in all, the platform can be said to have fully met the expectations set for it in earlier stages of development.

Chapter 7

Conclusion

In this chapter, there will be an examination of the project, through a summary of what was developed and presented throughout this thesis and a review of the level of fulfillment of the platform, while also addressing aspects left to be done or improved in the future. Also, there will be an additional section containing a brief personal overview of the project, its difficulties and achievements, and the quality of the work that was developed.

7.1 Summary of Fulfilled Work

The development of the project began through dialogues with stakeholders so as to define the problem at hand - the lack of interoperability among the applications, devices and services participating in integration tests during clinical trials - and the proposed solution - a data persistence platform that uses structuring standards to uniform the data and provide a centralized component for data exchange - as was discussed in Chapter 1. These discussions with stakeholders also allowed for more concrete requirements to be defined and further conceptualized, as well as supplemental planning for the solution's design, as documented in Chapter 4.

Alongside these procedures pertaining to Requirements Engineering, research regarding the topics of technology in the medical field, clinical trials, regulatory norms and healthcare data standards was executed so as to build a state of the art analysis and better understand the value of the proposed solution, which resulted in the findings presented in Chapter 2 and Chapter 3.

Taking all the preparation previously stated, the implementation of a prototype took place, using the HL7 FHIR standard. The prototype was developed in accordance to the prepared documentation and was also subjected, during and after its implementation, to tests and experiments that validated its conformance to the established requirements and its level of efficiency in performance. The steps taken in these stages of development were detailed in Chapter 5 and Chapter 6.

7.2 Review of Fulfilled Work

The platform managed to fully achieve the goals set for it early on, over its months of development. As of the writing of this thesis, its testing and revision by stakeholders and associated companies and infrastructures, as well as its integration in larger clinical contexts, is underway.

The FHIR Server proved to be resilient and fully capable of storing and restructuring the provided data, all while aiding in the coordination of the legal and ethical norms required when dealing with clinical data. The Management API was also successful in facilitating the input and output of data from the FHIR Server and in providing easy access to the server's capabilities.

Specifically, all the Use Cases described in Chapter 4.1.2 were fully implemented, with no issues detected, while also assuring conformity with the non-functional requirements of Chapter 4.1.3. The domain concepts shown in Chapter 4.1.4 and the architecture presented in Chapter 4.2.2 were completely respected during the implementation and are currently in effect, as of the writing of this thesis.

7.3 Future Work

While none of the presented use cases were left to be implemented and, as of the writing of this thesis, there are no additional functionalities or improvements that need to be imminently addressed, the only current concern for future work relates to the platform requiring occasional maintenance work and frequent monitoring to ensure its proper operability. Currently, third-party monitoring tools are being used to check in given time intervals for the availability of the platform, alerting the development team if the Management API or the FHIR Server become unavailable. It has been suggested that while using these tools is a viable solution for now, it might be of interest to design our own tools or protocols to alert for any disturbances to the platform's execution.

Moreover, as more companies and infrastructures join the project, there might be more pieces of Smart Health technology using the platform and, as such, there might be more clinical concepts to address and to prepare the Management API to receive and process (for example, new types of measurements). In such a case, new Use Cases will be created and the platform will reenter the development and testing stage, to accommodate these new functionalities. In relation to this, with increasing domain entities and concepts, there might be an interest in the adoption of Mapper classes to separate the business logic depicted in the service classes, from the restructuring of data, which would occur in these new classes. So far, this hasn't been demanded, but could prove to be beneficial for the efficiency of the API.

7.4 Personal Appreciation

A project of this nature, inserted in a crucial and heavily controlled environment, had to be very carefully planned, setup and reviewed. It involved many stakeholders and moving pieces, which lead to constant revisions and redesigns as to what the platform was intended to be and how it should act. Considering all of this, the final result proved to be successful and met the expectations set for it.

The most complex and arduous aspect of the project development was delving into the concepts of data structuring standards (FHIR) and the configuration and use of the HAPI FHIR Server. Understanding the inner workings of the server, the requirements for its proper functioning and its many quirks, was a long process that required many hours of research and testing.

Besides the tests and experiments shown in Chapter 6, there was constant communication with stakeholders and other entities that would make use of, or depend on the platform, to execute trials and assure that the platform was working in accordance with the needs and requirements of the project. The frequent contact among the various participants of the SMART-HEALTH-4-ALL initiative, executed through weekly meetings and online correspondence, helped fix any inconsistencies or issues that arose while integrating the various components of the project. This helped maintain the project in a due course and was only possible due to constant cooperation from everyone involved.

Bibliography

- Ayaz, Muhammad et al. (2021). "The Fast Health Interoperability Resources (FHIR) standard: systematic literature review of implementations, applications, challenges and opportunities". In: *JMIR medical informatics* 9.7, e21929.
- Badawi, Hawazin Faiz, Fedwa Laamarti, and Abdulmotaleb El Saddik (2019). "ISO/IEEE 11073 Personal Health Device (X73-PHD) Standards Compliant Systems: A Systematic Literature Review". In: *IEEE Access* 7, pp. 3062–3073. doi: 10.1109/ACCESS.2018.2886818.
- Bartolomei, JE and T Miller (2001). "Functional Analysis Systems Technique (FAST) as a group knowledge elicitation method for model building". In: *Proceeding of The 19th International Conference of the System Dynamics Society*.
- Bodenreider et al. (2018). "Recent developments in clinical terminologies—SNOMED CT, LOINC, and RxNorm". In: *Yearbook of medical informatics*. url: <https://www.thieme-connect.com/products/ejournals/pdf/10.1055/s-0038-1667077.pdf>.
- Borza, John (2011). "FAST diagrams: The foundation for creating effective function models". In: *Trizcon Detroit*.
- Bundle (2023). url: <https://build.fhir.org/bundle.html> (visited on 01/19/2023).
- Caristix HL7-Definition V2 reference site (2022). url: <https://hl7-definition.caristix.com/v2/> (visited on 12/05/2022).
- Carter, A. B. et al. (2020). "Use of LOINC for interoperability between organisations poses a risk to safety". In: url: [https://www.thelancet.com/journals/landig/article/PIIS2589-7500\(20\)30244-2/fulltext](https://www.thelancet.com/journals/landig/article/PIIS2589-7500(20)30244-2/fulltext).
- Dixon, James R (1999). "THE INTERNATIONAL CONFERENCE ON HARMONIZATION GOOD CLINICAL PRACTICE GUIDELINE". In: *Quality Assurance* 6.2, pp. 65–74. doi: 10.1080/105294199277860. eprint: <https://doi.org/10.1080/105294199277860>. url: <https://doi.org/10.1080/105294199277860>.
- Drenkhahn, Cora and Josef Ingenerf (June 2020). "The LOINC Content Model and Its Limitations of Usage in the Laboratory Domain". In: vol. 270. doi: 10.3233/SHTI200198.
- EvaluateMedTech (Sept. 2018). *EvaluateMedTech, World Preview 2018, Outlook to 2024*. <https://info.evaluategroup.com/rs/607-YGS-364/images/WPMT2018.pdf>.
- FHIR Observation (2022). url: <https://build.fhir.org/observation.html> (visited on 12/10/2022).
- FHIR Organization (2022). url: <https://build.fhir.org/organization.html> (visited on 12/10/2022).
- FHIR Overview - Developers (2022). url: <https://hl7.org/fhir/overview-dev.html> (visited on 12/10/2022).
- FHIR Publication (Version) History (2022). url: <http://hl7.org/fhir/directory.html> (visited on 12/09/2022).
- FHIR Resource Index (2022). url: <https://hl7.org/fhir/resourceindex.html> (visited on 12/09/2022).
- FHIR Server Types (2023). url: https://hapifhir.io/hapi-fhir/docs/server_plain/server_types.html (visited on 04/01/2023).

- Fraunhofer Portugal (2023). url: https://www.aicos.fraunhofer.pt/en/our_work/projects/smarthealth4all.html (visited on 01/10/2023).
- Friedman, Lawrence M et al. (2015). *Fundamentals of clinical trials*. Springer.
- Gamal, Barakat, and Rezk (2021). In: 114, p. 103670. issn: 1532-0464. doi: <https://doi.org/10.1016/j.jbi.2020.103670>. url: <https://www.sciencedirect.com/science/article/pii/S1532046420302987>.
- Goossen, William, and Laura Heermann Langford (Apr. 2014). "Exchanging care records using HL7 V3 care provision messages". In: *Journal of the American Medical Informatics Association : JAMIA* 21. doi: 10.1136/amiajnl-2013-002264.
- HAPI FHIR JPA Server (2023). url: <https://github.com/hapifhir/hapi-fhir-jpaserver-starter> (visited on 04/01/2023).
- HL7 CDA® Release 2 Product Suite (2022). url: http://www.hl7.org/implement/standards/product_brief.cfm?product_id=7 (visited on 12/06/2022).
- HL7 Standards (2022). url: <https://www.hl7.org/implement/standards/index.cfm?ref=nav> (visited on 12/05/2022).
- HL7 Version 2 Product Suite (2022). url: http://www.hl7.org/implement/standards/product_brief.cfm?product_id=185 (visited on 12/05/2022).
- HL7 Version 3 Product Suite (2022). url: http://www.hl7.org/implement/standards/product_brief.cfm?product_id=186 (visited on 12/06/2022).
- Huang, Chih-Yuan, and Hsin-Hsien Chen (Jan. 2019). "An Automatic Embedded Device Registration Procedure Based on the OGC SensorThings API". In: *Sensors* 19, p. 495. doi: 10.3390/s19030495.
- Hussain, Mohannad A, Steve G Langer, and Marc Kohli (2018). "Learning HL7 FHIR using the HAPI FHIR server and its use in medical imaging with the SIIM dataset". In: *Journal of digital imaging* 31.3, pp. 334–340.
- IEC 62304 (2023). url: <https://www.iso.org/standard/38421.html> (visited on 01/20/2023).
- Iroju et al. (Apr. 2013). "Interoperability in Healthcare: Benefits, Challenges and Resolutions". In: *International Journal of Innovation and Applied Studies* 3, pp. 2028–9324.
- ISO 13485:2016 (2023). url: <https://www.iso.org/standard/59752.html> (visited on 01/20/2023).
- Kasparick, Martin et al. (2015). "New IEEE 11073 standards for interoperable, networked point-of-care Medical Devices". In: *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 1721–1724. doi: 10.1109/EMBC.2015.7318709.
- Koen et al. (2001). "Fuzzy Front End: Effective Methods, Tools, and Techniques". In.
- Kruchten, Philippe (Nov. 1995). "The 4+1 View Model of Architecture". In: *IEEE Software* 12, pp. 45–50. doi: 10.1109/52.469759.
- Laamarti, Fedwa et al. (2020). "An ISO/IEEE 11073 Standardized Digital Twin Framework for Health and Well-Being in Smart Cities". In: *IEEE Access* 8, pp. 105950–105961. doi: 10.1109/ACCESS.2020.2999871.
- Lee, Sungkee and Hyoungho Do (2018). "Comparison and analysis of ISO/IEEE 11073, IHE PCD-01, and HL7 FHIR messages for personal health devices". In: *Healthcare informatics research* 24.1, pp. 46–52.
- Lehne, Moritz et al. (2019). "The Use of FHIR in Digital Health-A Review of the Scientific Literature." In: *GMDS* September, pp. 52–58.
- "Lei n.º 21/2014, de 16 de abril" (2014-04-16). In: *Diário da República n.º 75/2014 Série I*, pp. 2450–2465.

- "Lei n.º 58/2019, de 8 de agosto" (2019-08-08). In: *Diário da República n.º 151/2019 Série I*, pp. 3–40.
- LOINC Release Notes* (2022). url: <https://loinc.org/kb/loinc-release-notes/> (visited on 12/06/2022).
- Nicola, Susana and Eduarda Pinto Ferreira and J. J. Pinto Ferreira (2012). "International Journal of Information Technology Decision Making". In.
- OGC SensorThings API Part 1: Sensing Version 1.1* (2022). url: <https://docs.ogc.org/is/18-088/18-088.html> (visited on 12/07/2022).
- OGC SensorThings API Part 2 – Tasking Core* (2022). url: <http://docs.opengeospatial.org/is/17-079r1/17-079r1.html> (visited on 12/07/2022).
- Osterwalder, Alexander and Yves Pigneur (2013). "MODELLING VALUE PROPOSITIONS IN E-BUSINESS". In.
- Osterwalder et al. (2014). "Value Proposition Design: How to Create Products and Services Customers Want". In.
- Postman* (2023). url: <https://www.postman.com/> (visited on 02/01/2023).
- R* (2023). url: <https://www.r-project.org> (visited on 02/01/2023).
- "Regulamento n.º 798/2018, de 30 de novembro" (2018-11-30). In: *Diário da República n.º 231/2018 Série II*, pp. 32031–32032.
- "REGULATION (EU) 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL" (2016-05-04). In: *Official Journal of the European Union* L 119/1.
- "REGULATION (EU) 2017/745 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL" (2017-05-05). In: *Official Journal of the European Union* L 117/1.
- "REGULATION (EU) No 536/2014 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL" (2014-04-16). In: *Official Journal of the European Union* L 158/1.
- Rich, Nick and Matthias Holweg (2000). "Value analysis/value engineering". In: *Report produced for the EC funded project "INNOREGIO: dissemination of innovation and knowledge management techniques"*, Lean Enterprise Research Centre, Cardiff, UK.
- Rubio, Óscar J. et al. (2016). "Analysis of ISO/IEEE 11073 built-in security and its potential IHE-based extensibility". In: *Journal of Biomedical Informatics* 60, pp. 270–285. issn: 1532-0464. doi: <https://doi.org/10.1016/j.jbi.2016.02.006>. url: <https://www.sciencedirect.com/science/article/pii/S1532046416000289>.
- Santhanavanich, C. Kim T., and V. Coors. (2020). "Integration of Heterogeneous Coronavirus Disease COVID-19 Data Sources Using Ogc Sensorthings Api". In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*. url: <https://www.proquest.com/openview/3a89f290aa4e754ce27f3215573add8f/1?pq-origsite=gscholar&cbl=2037681>.
- Schulz et al. (2019). "Standards in healthcare data." *Fundamentals of Clinical Data Science*. In: url: <https://library.oapen.org/bitstream/handle/20.500.12657/22918/1007243.pdf>.
- SMART-HEALTH-4-ALL* (2023). url: <https://www.smarthealth4all.com/> (visited on 01/10/2023).
- Spring Framework* (2023). url: <https://spring.io/projects/spring-framework> (visited on 04/01/2023).
- Spronk (2007). *HL7 Message examples: version 2 and version 3*. Tech. rep. url: http://ringholm.com/docs/04300_en.htm.
- Tungpunkom (2015). "Steps in Conducting Systematic Review: It is Simple but Not That Easy." In.
- "WORLD MEDICAL ASSOCIATION DECLARATION OF HELSINKI Ethical Principles for Medical Research Involving Human Subjects" (2013-06-15). In.

-
- Yang, Zhe et al. (Mar. 2022). "Defining health data elements under the HL7 development framework for metadata management". In: *Journal of Biomedical Semantics* 13.1, p. 10. issn: 2041-1480. doi: 10.1186/s13326-022-00265-5. url: <https://doi.org/10.1186/s13326-022-00265-5>.
- Zeithaml, V.A (1988). "Consumer perceptions of price, quality and value: A means-end model and synthesis of evidence". In.

Appendix A

FHIR Resource Categories and Full Index

Foundation	Conformance <ul style="list-style-type: none"> CapabilityStatement N StructureDefinition N ImplementationGuide 1 SearchParameter 3 MessageDefinition 1 OperationDefinition N CompartmentDefinition 1 StructureMap 2 GraphDefinition 1 ExampleScenario 0 	Terminology <ul style="list-style-type: none"> CodeSystem N ValueSet N ConceptMap 3 NamingSystem 2 TerminologyCapabilities 0 	Security <ul style="list-style-type: none"> Provenance 3 AuditEvent 3 Consent 2 	Documents <ul style="list-style-type: none"> Composition 2 DocumentManifest 2 DocumentReference 3 CatalogEntry 0 	Other <ul style="list-style-type: none"> Basic 1 Binary N Bundle N Linkage 0 MessageHeader 4 OperationOutcome N Parameters N Subscription 3 SubscriptionStatus 0 SubscriptionTopic 0
	Base	Individuals <ul style="list-style-type: none"> Patient N Practitioner 3 PractitionerRole 2 RelatedPerson 2 Person 2 Group 1 	Entities #1 <ul style="list-style-type: none"> Organization 3 OrganizationAffiliation 0 HealthcareService 2 Endpoint 2 Location 3 	Entities #2 <ul style="list-style-type: none"> Substance 2 BiologicallyDerivedProduct 0 Device 2 DeviceMetric 1 NutritionProduct 0 	Workflow <ul style="list-style-type: none"> Task 2 Appointment 3 AppointmentResponse 3 Schedule 3 Slot 3 VerificationResult 0
Clinical		Summary <ul style="list-style-type: none"> AllergyIntolerance 3 AdverseEvent 0 Condition (Problem) 3 Procedure 3 FamilyMemberHistory 2 ClinicalImpression 0 DetectedIssue 1 	Diagnostics <ul style="list-style-type: none"> Observation N Media 1 DiagnosticReport 3 Specimen 2 BodyStructure 1 ImagingStudy 3 QuestionnaireResponse 3 MolecularSequence 1 	Medications <ul style="list-style-type: none"> MedicationRequest 3 MedicationAdministration 2 MedicationDispense 2 MedicationStatement 3 Medication 3 MedicationKnowledge 0 Immunization 3 ImmunizationEvaluation 0 ImmunizationRecommendation 1 	Care Provision <ul style="list-style-type: none"> CarePlan 2 CareTeam 2 Goal 2 ServiceRequest 2 NutritionOrder 2 VisionPrescription 2 RiskAssessment 1 RequestGroup 2
	Financial	Support <ul style="list-style-type: none"> Coverage 2 CoverageEligibilityRequest 2 CoverageEligibilityResponse 2 EnrollmentRequest 0 EnrollmentResponse 0 	Billing <ul style="list-style-type: none"> Claim 2 ClaimResponse 2 Invoice 0 	Payment <ul style="list-style-type: none"> PaymentNotice 2 PaymentReconciliation 2 	General <ul style="list-style-type: none"> Account 2 ChargeItem 0 ChargeItemDefinition 0 Contract 1 ExplanationOfBenefit 2 InsurancePlan 0
Specialized		Public Health & Research <ul style="list-style-type: none"> ResearchStudy 1 ResearchSubject 0 	Definitional Artifacts <ul style="list-style-type: none"> ActivityDefinition 3 DeviceDefinition 0 EventDefinition 0 ObservationDefinition 0 PlanDefinition 3 Questionnaire 3 SpecimenDefinition 0 	Evidence-Based Medicine <ul style="list-style-type: none"> Citation 0 Evidence 1 EvidenceReport 0 EvidenceVariable 1 	Quality Reporting & Testing <ul style="list-style-type: none"> Measure 3 MeasureReport 3 TestScript 2 TestReport 0

Figure A.1: FHIR Resource Categories and Full Index (*FHIR Resource Index 2022*).

Appendix B

Timed values of Performance Tests for Consecutive Requests

Table B.1: Timed values of Performance Tests for Consecutive Requests

Request #	Duration (ms)	Description
1	102	POST Clinical Trial
2	162	POST Patient
3	112	POST Patient
4	43	POST Patient
5	156	POST Patient
6	96	POST Patient
7	50	POST Clinical Trial
8	140	POST Patient
9	133	POST Patient
10	139	POST Patient
11	106	POST Patient
12	146	POST Clinical Trial
13	76	POST Patient
14	100	POST Patient
15	130	GET Patients
16	60	GET Patients
17	180	GET Patients
18	70	POST BioSensorData
19	86	POST BioSensorData
20	82	POST BioSensorData
21	191	POST BioSensorData
22	121	POST BioSensorData
23	183	POST BioSensorData
24	168	POST BioSensorData
25	111	POST BioSensorData
26	91	POST BioSensorData
27	149	POST BioSensorData
28	56	POST BioSensorData
29	199	POST BioSensorData
30	185	POST BioSensorData

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
31	181	POST BioSensorData
32	62	POST BioSensorData
33	156	POST BioSensorData
34	173	POST BioSensorData
35	182	POST BioSensorData
36	198	POST SleepBedroomData
37	113	POST SleepBedroomData
38	110	POST SleepBedroomData
39	190	POST SleepBedroomData
40	89	POST SleepBedroomData
41	64	POST SleepBedroomData
42	190	POST SleepBedroomData
43	104	POST SleepBedroomData
44	124	POST SleepBedroomData
45	51	POST SleepBedroomData
46	167	POST SleepBedroomData
47	107	POST SleepBedroomData
48	92	POST SleepBedroomData
49	198	POST SleepBedroomData
50	110	POST SleepBedroomData
51	113	POST SleepBedroomData
52	42	POST SleepBedroomData
53	142	POST SleepBedroomData
54	70	POST SleepBedroomData
55	134	POST SleepBedroomData
56	84	POST SleepBedroomData
57	137	POST SleepBedroomData
58	77	POST SleepBedroomData
59	46	POST SleepBedroomData
60	81	POST SleepBedroomData
61	98	POST SleepBedroomData
62	115	POST SleepBedroomData
63	141	POST SleepBedroomData
64	92	POST SleepBedroomData
65	42	POST SleepBedroomData
66	110	POST SleepBedroomData
67	165	POST SleepBedroomData
68	54	POST SleepBedroomData
69	137	POST SleepBedroomData
70	62	POST SleepBedroomData
71	180	POST SleepBedroomData
72	133	POST SleepBedroomData
73	57	POST SleepBedroomData
74	52	POST SleepBedroomData

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
75	58	POST SleepBedroomData
76	173	POST SleepBedroomData
77	130	POST SleepBedroomData
78	130	POST SleepBedroomData
79	143	POST SleepBedroomData
80	130	POST SleepBedroomData
81	53	POST SleepBedroomData
82	67	POST SleepBedroomData
83	80	POST SleepBedroomData
84	144	POST SleepBedroomData
85	156	POST SleepBedroomData
86	169	POST SleepBedroomData
87	202	POST SleepBedroomData
88	42	POST SleepBedroomData
89	200	POST SleepBedroomData
90	182	POST SleepBedroomData
91	140	POST SleepBedroomData
92	153	POST SleepBedroomData
93	159	POST SleepBedroomData
94	193	POST SleepBedroomData
95	106	POST SleepBedroomData
96	162	POST SleepBedroomData
97	200	POST SleepBedroomData
98	140	POST SleepBedroomData
99	135	POST SleepBedroomData
100	42	POST SleepBedroomData
101	60	POST SleepBedroomData
102	154	POST SleepBedroomData
103	173	POST SleepBedroomData
104	152	POST SleepBedroomData
105	152	POST SleepBedroomData
106	187	POST SleepBedroomData
107	164	POST SleepBedroomData
108	137	POST SleepBedroomData
109	52	POST SleepBedroomData
110	191	POST Clinical Trial
111	190	POST Clinical Trial
112	117	POST Patient
113	101	POST Patient
114	168	POST Patient
115	107	POST Patient
116	179	POST SleepBedroomData
117	83	POST SleepBedroomData
118	92	POST SleepBedroomData

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
119	52	POST SleepBedroomData
120	80	POST SleepBedroomData
121	105	POST SleepBedroomData
122	123	POST SleepBedroomData
123	116	POST SleepBedroomData
124	174	POST SleepBedroomData
125	117	POST SleepBedroomData
126	72	POST SleepBedroomData
127	132	POST SleepBedroomData
128	47	POST SleepBedroomData
129	69	POST SleepBedroomData
130	48	POST SleepBedroomData
131	94	POST SleepBedroomData
132	161	POST SleepBedroomData
133	200	POST SleepBedroomData
134	163	POST SleepBedroomData
135	101	POST SleepBedroomData
136	191	POST SleepBedroomData
137	88	POST BioSensorData
138	199	POST BioSensorData
139	107	POST BioSensorData
140	153	POST BioSensorData
141	190	POST BioSensorData
142	117	POST BioSensorData
143	60	POST BioSensorData
144	90	POST BioSensorData
145	179	POST BioSensorData
146	90	POST BioSensorData
147	145	POST Patient
148	162	POST Patient
149	62	POST Patient
150	83	POST Patient
151	199	POST Patient
152	184	POST Patient
153	117	POST Patient
154	100	POST Patient
155	151	POST Patient
156	129	POST Patient
157	191	POST Patient
158	191	POST Patient
159	171	POST BioSensorData
160	69	POST BioSensorData
161	71	POST SleepBedroomData
162	54	POST SleepBedroomData

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
163	134	POST BioSensorData
164	160	POST BioSensorData
165	185	POST SleepBedroomData
166	145	POST SleepBedroomData
167	82	POST BioSensorData
168	73	POST BioSensorData
169	57	POST SleepBedroomData
170	70	POST SleepBedroomData
171	147	POST BioSensorData
172	40	POST BioSensorData
173	52	POST SleepBedroomData
174	51	POST SleepBedroomData
175	206	POST BioSensorData
176	120	POST BioSensorData
177	199	POST SleepBedroomData
178	112	POST SleepBedroomData
179	184	POST SleepBedroomData
180	185	POST BioSensorData
181	77	POST BioSensorData
182	116	POST SleepBedroomData
183	82	POST SleepBedroomData
184	130	POST BioSensorData
185	186	POST BioSensorData
186	111	POST SleepBedroomData
187	137	POST SleepBedroomData
188	59	POST BioSensorData
189	171	POST BioSensorData
190	57	POST SleepBedroomData
191	170	POST SleepBedroomData
192	160	POST BioSensorData
193	147	POST SleepBedroomData
194	70	POST BioSensorData
195	111	POST BioSensorData
196	183	POST SleepBedroomData
197	98	POST SleepBedroomData
198	108	POST BioSensorData
199	107	POST BioSensorData
200	202	POST SleepBedroomData
201	55	POST SleepBedroomData
202	163	POST BioSensorData
203	86	POST BioSensorData
204	204	POST SleepBedroomData
205	114	POST SleepBedroomData
206	202	POST BioSensorData

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
207	185	POST SleepBedroomData
208	82	POST BioSensorData
209	94	POST BioSensorData
210	134	POST SleepBedroomData
211	196	POST SleepBedroomData
212	208	POST BioSensorData
213	143	POST BioSensorData
214	206	POST SleepBedroomData
215	142	POST SleepBedroomData
216	192	POST BioSensorData
217	111	POST BioSensorData
218	127	POST SleepBedroomData
219	210	POST SleepBedroomData
220	104	POST BioSensorData
221	114	POST BioSensorData
222	81	POST SleepBedroomData
223	56	POST SleepBedroomData
224	87	POST SleepBedroomData
225	142	POST Patient
226	53	POST Patient
227	79	POST Patient
228	116	POST Patient
229	157	POST Patient
230	75	POST Patient
231	87	POST Patient
232	137	POST Patient
233	50	POST BioSensorData
234	197	POST BioSensorData
235	181	POST BioSensorData
236	179	POST BioSensorData
237	121	POST BioSensorData
238	155	POST BioSensorData
239	185	POST BioSensorData
240	207	POST BioSensorData
241	73	POST BioSensorData
242	193	POST BioSensorData
243	203	POST BioSensorData
244	54	POST BioSensorData
245	183	POST BioSensorData
246	163	POST BioSensorData
247	48	POST BioSensorData
248	92	POST BioSensorData
249	53	POST BioSensorData
250	138	POST SleepBedroomData

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
251	127	POST SleepBedroomData
252	74	POST SleepBedroomData
253	149	POST SleepBedroomData
254	108	POST SleepBedroomData
255	206	POST SleepBedroomData
256	101	POST SleepBedroomData
257	124	POST SleepBedroomData
258	64	POST SleepBedroomData
259	153	POST SleepBedroomData
260	56	POST SleepBedroomData
261	67	POST SleepBedroomData
262	162	POST SleepBedroomData
263	105	POST SleepBedroomData
264	170	POST SleepBedroomData
265	113	POST SleepBedroomData
266	151	POST SleepBedroomData
267	169	POST SleepBedroomData
268	136	POST SleepBedroomData
269	191	POST SleepBedroomData
270	55	POST SleepBedroomData
271	71	POST SleepBedroomData
272	90	POST SleepBedroomData
273	102	POST SleepBedroomData
274	67	POST SleepBedroomData
275	121	POST SleepBedroomData
276	138	POST SleepBedroomData
277	150	POST SleepBedroomData
278	151	POST SleepBedroomData
279	120	POST SleepBedroomData
280	120	POST SleepBedroomData
281	49	POST SleepBedroomData
282	157	POST SleepBedroomData
283	40	POST SleepBedroomData
284	43	POST SleepBedroomData
285	42	POST SleepBedroomData
286	54	POST SleepBedroomData
287	44	POST SleepBedroomData
288	69	POST SleepBedroomData
289	152	POST SleepBedroomData
290	60	POST SleepBedroomData
291	62	POST SleepBedroomData
292	199	POST SleepBedroomData
293	191	POST SleepBedroomData
294	93	POST SleepBedroomData

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
295	198	POST SleepBedroomData
296	64	POST SleepBedroomData
297	192	POST SleepBedroomData
298	181	POST SleepBedroomData
299	84	POST SleepBedroomData
300	148	POST SleepBedroomData
301	73	POST SleepBedroomData
302	73	POST Clinical Trial
303	74	POST Clinical Trial
304	46	POST Clinical Trial
305	90	POST Clinical Trial
306	137	POST Clinical Trial
307	176	POST Clinical Trial
308	182	POST Patient
309	128	POST Patient
310	127	POST Patient
311	149	POST Patient
312	54	POST Patient
313	198	POST Patient
314	90	POST Patient
315	122	POST Patient
316	118	POST Patient
317	59	POST Patient
318	41	POST Patient
319	177	POST Patient
320	208	POST Patient
321	181	POST Patient
322	160	POST Patient
323	46	POST Patient
324	69	POST Patient
325	84	POST Patient
326	166	POST Patient
327	200	POST Patient
328	50	POST Patient
329	112	POST Patient
330	51	POST Patient
331	161	POST Patient
332	199	POST Patient
333	54	POST Patient
334	90	POST BioSensorData
335	116	POST BioSensorData
336	200	POST BioSensorData
337	188	POST BioSensorData
338	178	POST BioSensorData

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
339	139	POST BioSensorData
340	146	POST BioSensorData
341	79	POST BioSensorData
342	112	POST BioSensorData
343	92	POST BioSensorData
344	130	POST BioSensorData
345	201	POST BioSensorData
346	86	POST BioSensorData
347	141	POST BioSensorData
348	195	POST BioSensorData
349	56	POST BioSensorData
350	108	POST BioSensorData
351	119	POST BioSensorData
352	153	POST BioSensorData
353	104	POST BioSensorData
354	50	POST BioSensorData
355	202	POST BioSensorData
356	117	POST BioSensorData
357	43	POST BioSensorData
358	161	POST BioSensorData
359	128	POST BioSensorData
360	131	POST BioSensorData
361	70	POST BioSensorData
362	68	POST BioSensorData
363	40	POST BioSensorData
364	198	POST BioSensorData
365	87	POST BioSensorData
366	190	POST BioSensorData
367	128	POST BioSensorData
368	126	POST BioSensorData
369	76	POST BioSensorData
370	152	POST BioSensorData
371	162	POST BioSensorData
372	198	POST BioSensorData
373	206	POST BioSensorData
374	78	POST BioSensorData
375	168	POST BioSensorData
376	63	POST BioSensorData
377	129	POST BioSensorData
378	159	POST BioSensorData
379	103	POST BioSensorData
380	149	POST BioSensorData
381	86	POST BioSensorData
382	140	POST BioSensorData

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
383	45	POST BioSensorData
384	204	POST BioSensorData
385	140	POST BioSensorData
386	97	POST BioSensorData
387	70	POST BioSensorData
388	141	POST BioSensorData
389	147	POST BioSensorData
390	65	POST BioSensorData
391	195	POST BioSensorData
392	110	POST BioSensorData
393	148	POST BioSensorData
394	71	POST BioSensorData
395	178	POST BioSensorData
396	54	POST BioSensorData
397	110	POST BioSensorData
398	143	POST BioSensorData
399	134	POST BioSensorData
400	188	GET BioSensorData of Patient
401	197	GET BioSensorData of Patient
402	177	GET BioSensorData of Patient
403	54	GET BioSensorData of Patient
404	166	GET BioSensorData of Patient
405	144	GET BioSensorData of Patient
406	99	GET BioSensorData of Patient
407	162	GET BioSensorData of Patient
408	71	GET BioSensorData of Patient
409	52	POST Patient
410	62	POST Patient
411	72	POST Patient
412	75	POST Patient
413	167	POST Patient
414	111	POST Patient
415	186	POST Patient
416	85	POST Patient
417	187	POST Patient
418	103	POST Patient
419	105	POST Patient
420	57	POST Patient
421	172	POST Patient
422	143	POST BioSensorData
423	128	POST BioSensorData
424	167	POST SleepBedroomData
425	87	POST SleepBedroomData
426	84	POST BioSensorData

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
427	139	POST BioSensorData
428	134	POST SleepBedroomData
429	67	POST SleepBedroomData
430	155	POST BioSensorData
431	180	POST BioSensorData
432	92	POST SleepBedroomData
433	185	POST SleepBedroomData
434	106	POST BioSensorData
435	118	POST BioSensorData
436	110	POST SleepBedroomData
437	163	POST SleepBedroomData
438	57	POST BioSensorData
439	170	POST BioSensorData
440	105	POST SleepBedroomData
441	107	POST SleepBedroomData
442	103	POST SleepBedroomData
443	162	POST BioSensorData
444	140	POST BioSensorData
445	122	POST SleepBedroomData
446	173	POST SleepBedroomData
447	210	POST BioSensorData
448	130	POST BioSensorData
449	121	POST SleepBedroomData
450	72	POST SleepBedroomData
451	206	POST BioSensorData
452	136	POST BioSensorData
453	199	POST SleepBedroomData
454	139	POST SleepBedroomData
455	106	POST BioSensorData
456	102	POST SleepBedroomData
457	73	POST BioSensorData
458	60	POST BioSensorData
459	68	POST SleepBedroomData
460	42	POST SleepBedroomData
461	196	POST BioSensorData
462	84	POST BioSensorData
463	194	POST SleepBedroomData
464	123	POST SleepBedroomData
465	41	POST BioSensorData
466	163	POST BioSensorData
467	188	POST SleepBedroomData
468	206	POST SleepBedroomData
469	148	POST Patient
470	165	POST Patient

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
471	94	POST Patient
472	99	POST Patient
473	95	POST Patient
474	69	POST Patient
475	128	POST Patient
476	180	POST Patient
477	162	POST Patient
478	123	POST Patient
479	116	POST Patient
480	52	POST Patient
481	100	POST BioSensorData
482	131	POST BioSensorData
483	94	POST SleepBedroomData
484	209	POST SleepBedroomData
485	178	POST BioSensorData
486	127	POST BioSensorData
487	46	POST SleepBedroomData
488	63	POST SleepBedroomData
489	164	POST BioSensorData
490	165	POST BioSensorData
491	203	POST SleepBedroomData
492	86	POST SleepBedroomData
493	138	POST BioSensorData
494	177	POST BioSensorData
495	68	POST SleepBedroomData
496	54	POST SleepBedroomData
497	168	POST BioSensorData
498	69	POST BioSensorData
499	126	POST SleepBedroomData
500	187	POST SleepBedroomData
501	192	POST SleepBedroomData
502	209	POST BioSensorData
503	82	POST BioSensorData
504	147	POST SleepBedroomData
505	124	POST SleepBedroomData
506	80	POST BioSensorData
507	158	POST BioSensorData
508	87	POST SleepBedroomData
509	147	POST SleepBedroomData
510	133	POST BioSensorData
511	171	POST BioSensorData
512	124	POST SleepBedroomData
513	82	POST SleepBedroomData
514	153	POST BioSensorData

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
515	75	POST SleepBedroomData
516	44	POST BioSensorData
517	139	POST BioSensorData
518	56	POST SleepBedroomData
519	210	POST SleepBedroomData
520	126	POST BioSensorData
521	183	POST BioSensorData
522	201	POST SleepBedroomData
523	145	POST SleepBedroomData
524	189	POST BioSensorData
525	84	POST BioSensorData
526	73	POST SleepBedroomData
527	189	POST SleepBedroomData
528	131	GET Patients
529	170	GET BioSensorData of Patient
530	172	GET BioSensorData of Patient
531	63	GET BioSensorData of Patient
532	202	GET BioSensorData of Patient
533	209	GET BioSensorData of Patient
534	159	GET BioSensorData of Patient
535	96	GET BioSensorData of Patient
536	195	GET BioSensorData of Patient
537	163	GET BioSensorData of Patient
538	47	GET BioSensorData of Patient
539	150	GET BioSensorData of Patient
540	147	GET BioSensorData of Patient
541	124	GET BioSensorData of Patient
542	75	GET BioSensorData of Patient
543	48	GET BioSensorData of Patient
544	163	GET BioSensorData of Patient
545	127	GET BioSensorData of Patient
546	138	GET BioSensorData of Patient
547	150	GET BioSensorData of Patient
548	112	GET BioSensorData of Patient
549	54	GET BioSensorData of Patient
550	64	GET BioSensorData of Patient
551	70	POST BioSensorData
552	177	POST BioSensorData
553	44	POST SleepBedroomData
554	171	POST SleepBedroomData
555	132	POST BioSensorData
556	114	POST SleepBedroomData
557	54	POST BioSensorData
558	79	POST BioSensorData

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
559	54	POST SleepBedroomData
560	198	POST SleepBedroomData
561	184	POST BioSensorData
562	41	POST BioSensorData
563	63	POST SleepBedroomData
564	112	POST SleepBedroomData
565	78	POST BioSensorData
566	149	POST BioSensorData
567	181	POST SleepBedroomData
568	108	POST SleepBedroomData
569	179	GET Patients
570	60	GET BioSensorData of Patient
571	153	GET BioSensorData of Patient
572	70	GET BioSensorData of Patient
573	154	GET BioSensorData of Patient
574	167	GET BioSensorData of Patient
575	134	GET BioSensorData of Patient
576	80	GET BioSensorData of Patient
577	118	GET BioSensorData of Patient
578	40	GET BioSensorData of Patient
579	42	GET BioSensorData of Patient
580	210	GET BioSensorData of Patient
581	154	GET BioSensorData of Patient
582	127	GET BioSensorData of Patient
583	121	POST BioSensorData
584	202	POST BioSensorData
585	113	POST SleepBedroomData
586	83	POST SleepBedroomData
587	160	POST BioSensorData
588	91	POST SleepBedroomData
589	64	POST BioSensorData
590	134	POST BioSensorData
591	178	POST SleepBedroomData
592	163	POST SleepBedroomData
593	177	POST BioSensorData
594	59	POST BioSensorData
595	163	POST SleepBedroomData
596	145	POST SleepBedroomData
597	84	POST BioSensorData
598	46	POST BioSensorData
599	45	POST SleepBedroomData
600	67	POST SleepBedroomData
601	155	GET Patients
602	186	GET BioSensorData of Patient

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
603	132	GET BioSensorData of Patient
604	121	GET BioSensorData of Patient
605	158	GET BioSensorData of Patient
606	66	GET BioSensorData of Patient
607	200	GET BioSensorData of Patient
608	209	GET BioSensorData of Patient
609	92	GET BioSensorData of Patient
610	72	GET BioSensorData of Patient
611	189	GET BioSensorData of Patient
612	100	GET BioSensorData of Patient
613	197	GET BioSensorData of Patient
614	65	GET BioSensorData of Patient
615	78	POST Clinical Trial
616	105	POST Patient
617	106	POST Patient
618	170	POST Patient
619	143	POST Patient
620	198	POST Patient
621	160	POST Clinical Trial
622	69	POST Patient
623	191	POST Patient
624	138	POST Patient
625	100	POST Patient
626	47	POST Clinical Trial
627	165	POST Patient
628	86	POST Patient
629	171	GET Patients
630	165	GET Patients
631	154	GET Patients
632	117	POST BioSensorData
633	98	POST BioSensorData
634	134	POST BioSensorData
635	40	POST BioSensorData
636	50	POST BioSensorData
637	162	POST BioSensorData
638	89	POST BioSensorData
639	75	POST BioSensorData
640	92	POST BioSensorData
641	116	POST BioSensorData
642	207	POST BioSensorData
643	42	POST BioSensorData
644	74	POST BioSensorData
645	114	POST BioSensorData
646	185	POST BioSensorData

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
647	194	POST BioSensorData
648	201	POST BioSensorData
649	173	POST BioSensorData
650	102	POST SleepBedroomData
651	198	POST SleepBedroomData
652	186	POST SleepBedroomData
653	169	POST SleepBedroomData
654	47	POST SleepBedroomData
655	109	POST SleepBedroomData
656	72	POST SleepBedroomData
657	189	POST SleepBedroomData
658	116	POST SleepBedroomData
659	106	POST SleepBedroomData
660	148	POST SleepBedroomData
661	131	POST SleepBedroomData
662	97	POST SleepBedroomData
663	207	POST SleepBedroomData
664	169	POST SleepBedroomData
665	73	POST SleepBedroomData
666	79	POST SleepBedroomData
667	46	POST SleepBedroomData
668	79	POST SleepBedroomData
669	194	POST SleepBedroomData
670	147	POST SleepBedroomData
671	186	POST SleepBedroomData
672	156	POST SleepBedroomData
673	134	POST SleepBedroomData
674	185	POST SleepBedroomData
675	128	POST SleepBedroomData
676	175	POST SleepBedroomData
677	54	POST SleepBedroomData
678	145	POST SleepBedroomData
679	118	POST SleepBedroomData
680	75	POST SleepBedroomData
681	66	POST SleepBedroomData
682	97	POST SleepBedroomData
683	108	POST SleepBedroomData
684	108	POST SleepBedroomData
685	188	POST SleepBedroomData
686	75	POST SleepBedroomData
687	173	POST SleepBedroomData
688	65	POST SleepBedroomData
689	67	POST SleepBedroomData
690	95	POST SleepBedroomData

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
691	197	POST SleepBedroomData
692	77	POST SleepBedroomData
693	123	POST SleepBedroomData
694	65	POST SleepBedroomData
695	173	POST SleepBedroomData
696	161	POST SleepBedroomData
697	162	POST SleepBedroomData
698	175	POST SleepBedroomData
699	192	POST SleepBedroomData
700	187	POST SleepBedroomData
701	40	POST SleepBedroomData
702	96	POST SleepBedroomData
703	98	POST SleepBedroomData
704	109	POST SleepBedroomData
705	117	POST SleepBedroomData
706	84	POST SleepBedroomData
707	159	POST SleepBedroomData
708	41	POST SleepBedroomData
709	51	POST SleepBedroomData
710	112	POST SleepBedroomData
711	123	POST SleepBedroomData
712	57	POST SleepBedroomData
713	98	POST SleepBedroomData
714	181	POST SleepBedroomData
715	177	POST SleepBedroomData
716	52	POST SleepBedroomData
717	177	POST SleepBedroomData
718	170	POST SleepBedroomData
719	74	GET BioSensorData of Patient
720	115	GET BioSensorData of Patient
721	176	GET BioSensorData of Patient
722	130	GET BioSensorData of Patient
723	99	GET BioSensorData of Patient
724	150	GET BioSensorData of Patient
725	125	GET BioSensorData of Patient
726	191	GET BioSensorData of Patient
727	82	GET BioSensorData of Patient
728	131	GET BioSensorData of Patient
729	205	GET BioSensorData of Patient
730	90	GET BioSensorData of Patient
731	83	GET BioSensorData of Patient
732	136	GET BioSensorData of Patient
733	171	GET BioSensorData of Patient
734	135	GET BioSensorData of Patient

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
735	128	GET BioSensorData of Patient
736	150	GET BioSensorData of Patient
737	175	GET BioSensorData of Patient
738	200	GET BioSensorData of Patient
739	98	GET BioSensorData of Patient
740	75	GET BioSensorData of Patient
741	98	GET BioSensorData of Patient
742	158	GET BioSensorData of Patient
743	160	GET BioSensorData of Patient
744	117	GET BioSensorData of Patient
745	124	POST BioSensorData
746	165	POST BioSensorData
747	48	POST SleepBedroomData
748	123	POST SleepBedroomData
749	113	POST BioSensorData
750	102	POST SleepBedroomData
751	40	POST BioSensorData
752	193	POST BioSensorData
753	98	POST SleepBedroomData
754	95	POST SleepBedroomData
755	123	POST BioSensorData
756	124	POST BioSensorData
757	114	POST SleepBedroomData
758	84	POST SleepBedroomData
759	192	POST BioSensorData
760	138	POST BioSensorData
761	206	POST SleepBedroomData
762	74	POST SleepBedroomData
763	176	GET Patients
764	148	GET BioSensorData of Patient
765	158	GET BioSensorData of Patient
766	193	GET BioSensorData of Patient
767	163	GET BioSensorData of Patient
768	67	GET BioSensorData of Patient
769	141	GET BioSensorData of Patient
770	97	GET BioSensorData of Patient
771	131	GET BioSensorData of Patient
772	108	GET BioSensorData of Patient
773	97	GET BioSensorData of Patient
774	151	GET BioSensorData of Patient
775	98	GET BioSensorData of Patient
776	95	GET BioSensorData of Patient
777	79	POST Clinical Trial
778	82	POST Patient

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
779	100	POST Patient
780	40	POST Patient
781	92	POST Patient
782	114	POST Patient
783	116	POST Clinical Trial
784	91	POST Patient
785	82	POST Patient
786	164	POST Patient
787	135	POST Patient
788	77	POST Clinical Trial
789	104	POST Patient
790	184	POST Patient
791	76	GET Patients
792	133	GET Patients
793	133	GET Patients
794	99	POST BioSensorData
795	57	POST BioSensorData
796	178	POST BioSensorData
797	108	POST BioSensorData
798	130	POST BioSensorData
799	148	POST BioSensorData
800	100	POST BioSensorData
801	70	POST BioSensorData
802	107	POST BioSensorData
803	42	POST BioSensorData
804	159	POST BioSensorData
805	84	POST BioSensorData
806	201	POST BioSensorData
807	57	POST SleepBedroomData
808	127	POST SleepBedroomData
809	51	POST BioSensorData
810	160	POST SleepBedroomData
811	64	POST BioSensorData
812	54	POST BioSensorData
813	127	POST SleepBedroomData
814	51	POST SleepBedroomData
815	156	POST BioSensorData
816	175	POST BioSensorData
817	168	POST SleepBedroomData
818	95	POST SleepBedroomData
819	97	POST BioSensorData
820	144	POST BioSensorData
821	192	POST SleepBedroomData
822	169	POST SleepBedroomData

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
823	186	GET Patients
824	162	GET BioSensorData of Patient
825	208	GET BioSensorData of Patient
826	155	GET BioSensorData of Patient
827	204	GET BioSensorData of Patient
828	47	GET BioSensorData of Patient
829	97	GET BioSensorData of Patient
830	73	GET BioSensorData of Patient
831	70	GET BioSensorData of Patient
832	64	GET BioSensorData of Patient
833	45	GET BioSensorData of Patient
834	90	GET BioSensorData of Patient
835	127	GET BioSensorData of Patient
836	102	GET BioSensorData of Patient
837	135	POST Clinical Trial
838	94	POST Patient
839	75	POST Patient
840	90	POST Patient
841	143	POST Patient
842	96	POST Patient
843	139	POST Clinical Trial
844	87	POST Patient
845	68	POST Patient
846	105	POST Patient
847	154	POST Patient
848	75	POST Clinical Trial
849	77	POST Patient
850	142	POST Patient
851	74	GET Patients
852	54	GET Patients
853	117	GET Patients
854	175	POST BioSensorData
855	208	POST BioSensorData
856	190	POST BioSensorData
857	169	POST BioSensorData
858	163	POST BioSensorData
859	196	POST BioSensorData
860	63	POST BioSensorData
861	184	POST BioSensorData
862	80	POST BioSensorData
863	94	POST BioSensorData
864	46	POST BioSensorData
865	151	POST BioSensorData
866	130	POST BioSensorData

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
867	155	POST SleepBedroomData
868	144	POST SleepBedroomData
869	88	POST BioSensorData
870	160	POST SleepBedroomData
871	210	POST BioSensorData
872	145	POST BioSensorData
873	165	POST SleepBedroomData
874	190	POST SleepBedroomData
875	53	POST BioSensorData
876	100	POST BioSensorData
877	150	POST SleepBedroomData
878	167	POST SleepBedroomData
879	74	POST BioSensorData
880	143	POST BioSensorData
881	168	POST SleepBedroomData
882	47	POST SleepBedroomData
883	120	GET Patients
884	42	GET BioSensorData of Patient
885	45	GET BioSensorData of Patient
886	185	GET BioSensorData of Patient
887	89	GET BioSensorData of Patient
888	83	GET BioSensorData of Patient
889	54	GET BioSensorData of Patient
890	176	GET BioSensorData of Patient
891	149	GET BioSensorData of Patient
892	66	GET BioSensorData of Patient
893	106	GET BioSensorData of Patient
894	179	GET BioSensorData of Patient
895	169	GET BioSensorData of Patient
896	180	GET BioSensorData of Patient
897	96	POST Clinical Trial
898	57	POST Patient
899	190	POST Patient
900	195	POST Patient
901	78	POST Patient
902	92	POST Patient
903	46	POST Clinical Trial
904	196	POST Patient
905	62	POST Patient
906	120	POST Patient
907	92	POST Patient
908	137	POST Clinical Trial
909	139	POST Patient
910	128	POST Patient

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
911	139	GET Patients
912	169	GET Patients
913	132	GET Patients
914	81	POST BioSensorData
915	188	POST BioSensorData
916	90	POST BioSensorData
917	194	POST BioSensorData
918	89	POST BioSensorData
919	146	POST BioSensorData
920	69	POST BioSensorData
921	143	POST BioSensorData
922	125	POST BioSensorData
923	101	POST BioSensorData
924	92	POST BioSensorData
925	209	GET BioSensorData of Patient
926	129	GET BioSensorData of Patient
927	207	GET BioSensorData of Patient
928	177	GET BioSensorData of Patient
929	94	GET BioSensorData of Patient
930	162	GET BioSensorData of Patient
931	121	GET BioSensorData of Patient
932	141	GET BioSensorData of Patient
933	169	GET BioSensorData of Patient
934	134	GET BioSensorData of Patient
935	116	GET BioSensorData of Patient
936	67	GET BioSensorData of Patient
937	61	GET BioSensorData of Patient
938	107	GET BioSensorData of Patient
939	94	GET BioSensorData of Patient
940	117	GET BioSensorData of Patient
941	134	GET BioSensorData of Patient
942	178	GET BioSensorData of Patient
943	134	GET BioSensorData of Patient
944	152	GET BioSensorData of Patient
945	183	GET BioSensorData of Patient
946	141	GET BioSensorData of Patient
947	99	GET BioSensorData of Patient
948	198	GET BioSensorData of Patient
949	183	GET BioSensorData of Patient
950	102	GET BioSensorData of Patient
951	201	GET BioSensorData of Patient
952	83	GET BioSensorData of Patient
953	146	GET BioSensorData of Patient
954	203	GET BioSensorData of Patient

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
955	84	GET BioSensorData of Patient
956	116	GET BioSensorData of Patient
957	92	POST Clinical Trial
958	110	POST Patient
959	42	POST Patient
960	122	POST Patient
961	205	POST Patient
962	202	POST Patient
963	100	POST Clinical Trial
964	115	POST Patient
965	123	POST Patient
966	72	POST Patient
967	190	POST Patient
968	44	POST Clinical Trial
969	51	POST Patient
970	109	POST Patient
971	111	GET Patients
972	133	GET Patients
973	198	GET Patients
974	72	POST BioSensorData
975	194	POST BioSensorData
976	133	POST BioSensorData
977	130	POST BioSensorData
978	91	POST BioSensorData
979	125	POST BioSensorData
980	188	POST BioSensorData
981	158	POST BioSensorData
982	61	POST BioSensorData
983	159	POST BioSensorData
984	131	POST BioSensorData
985	144	POST BioSensorData
986	200	POST BioSensorData
987	143	POST SleepBedroomData
988	126	POST SleepBedroomData
989	119	POST BioSensorData
990	68	POST SleepBedroomData
991	121	POST BioSensorData
992	89	POST BioSensorData
993	68	GET BioSensorData of Patient
994	144	GET BioSensorData of Patient
995	151	GET BioSensorData of Patient
996	80	GET BioSensorData of Patient
997	58	GET BioSensorData of Patient
998	89	GET BioSensorData of Patient

Continued on next page

Table B.1 – continued from previous page

Request #	Duration (ms)	Description
999	124	GET BioSensorData of Patient
1000	59	GET BioSensorData of Patient