



SNOMED on FHIR

**Transmission of clinical data with the Fast
Healthcare Interoperability
Resources protocol (HL7-FHIR) utilizing
Systematized Nomenclature of
Medicine - Clinical Terms (SNOMED-CT)**

Waqar Hassan

Supervisors

Associate Professor Jan Pettersen Nytnun

Associate Professor Martin Wulf Gerdes

SNOMED on FHIR

Transmission of clinical data with the Fast Healthcare Interoperability Resources protocol (HL7-FHIR) utilizing Systematized Nomenclature of Medicine - Clinical Terms (SNOMED-CT)

AUTHOR: WAQAR HASSAN

SUPERVISOR

Associate Professor Jan Pettersen Nytnun
Associate Professor Martin Wulf Gerdes

University of Agder, [2019]

Faculty of Engineering and Sciences
Department of Information and Communication Tecgnology

Abstract

The exchange of information at the level of semantic interoperability needs some information model and clinical terminology work together. Electronic records of the patient must include the terminologies in order to give the decision support to the practitioner when needed. Due to the lack of common standards for data structure and data sharing creates the problem to interact and share information with different applications.

Our main goal in this thesis is transferring the SNOMED-CT (Systematized Nomenclature of Medicine Clinical Terms) terminologies using HL7 FHIR (Fast Healthcare Interoperability Resources). Our first step is to setup the HAPI FHIR server in which we can store the clinical information. The next step is to create a web-application which defines the HL7 FHIR profiles for our system. Then implement these profiles on the RESTful FHIR server where all the clinical information is stored in the FHIR resources. The last step is to make the front-end and back-end services to support the RESTful HAPI FHIR server.

As proof, we have developed a web application that has capabilities to create the FHIR profiles, transfer the SNOMED-CT terminologies and gives the ability to search the SNOMED-CT and FHIR resources.

Keywords: SNOMED, HL7, FHIR, RESTful,

Preface

This thesis work is done in order to fulfill the requirement of IKT 590 Masters thesis course 30 ECT at the faculty of engineering and sciences, department of information and communication technology ICT at the University of Agder, Grimstad, Norway.

The work is started from 16th January 2019 and ended on 10th June 2019. The aim of this work is mapping the SNOMED-CT terminologies on the HL7 FHIR resources and provide the RESTful services to access the vital-signs. I would like to express my sincere gratitude to my supervisors, Associate Professor Jan Pettersen Nytnun and Associate Professor Martin Gerdes for the continuous support of my Master study and research, for all the help, motivation, enthusiasm, and immense knowledge through the period of thesis work.

Special thanks to parents for their prayers who are my source of enthusiasm.

I would like to thank all my classmates and friends.

Waqar Hassan

10th June 2019

University of Agder

Grimstad, Norway

HL7 Health Level Seven

SNOMED-CT Systemized Nomenclature of Medicines Clinical Terminologies

NDE Norwegian Directorate of E-Health

EHR Electronic Health Record

NUFA Academic committee for health and architecture in Norway

XML Extensible Markup Language

IHTSDO International Health Terminology Standard Organization

NIH National Institute of Health

XDS Cross-Enterprise Document Sharing

FHIR Fast Health Inseparability Resources

UCUM The Unified Code for Units of Measure

API Application Program Interface

CAP College of American Pathologists

NHS National Health Service

UHN University Health Network

NIH National Institute of Health

AMR Automated Medical Records

Contents

Abstract	i
Acknowledgements	i
List of Abbreviations	iv
1 Introduction	1
1.1 Background	2
1.2 Motivation	5
1.3 Research Questions	5
1.4 Proposed Solution	5
1.4.1 Developing a Restful services	6
1.4.2 Interaction of Web application with Restful services	6
1.5 Report Structure	7
2 Theoretical Background and Related Work	9
2.1 Electronic Health Record EHR	9
2.2 Health Level Seven International (HL7)	11
2.2.1 Evolution of HL7	12
2.3 Fast Healthcare Interoperability Resources (FHIR)	14

2.3.1	HL7 FHIR Release 4	15
2.3.2	Structure of FHIR	19
2.3.3	FHIR Composition Framework	20
2.3.4	The Four Paradigm	20
2.3.5	FHIR RESTful API	20
2.4	FHIR Concepts Need to Understand	21
2.4.1	FHIR Profiling	22
2.4.2	FHIR Extensions	23
2.4.3	FHIR Terminology Module	25
2.5	Focus on FHIR Resources	26
2.5.1	Patient Resource	27
2.5.2	The Observation Resource	28
2.5.3	Practitioner Resource	31
2.5.4	Relationship between Resources	31
2.6	Importance of Clinical Terminologies	32
2.6.1	SNOMED-CT	32
2.6.2	SNOMED in Norway	33
2.6.3	Origins of SNOMED	33
2.6.4	Structure of SNOMED	34
2.7	FHIR with SNOMED	36
2.8	Architecture of HAPI	37
2.9	Technologies	38
2.9.1	JSON	38
2.9.2	Node.js	38

2.9.3	Web Development	39
2.9.4	Apache Tomcat Server	39
3	System Design	40
3.1	Client Web Application Architecture	41
3.2	Used Tools	41
3.3	Defining HL7 FHIR Structure for profile	42
3.4	Resource Handling	44
3.4.1	Query for SNOMED-CT	44
4	Solution Deployment pre-requisite	46
4.1	HAPI FHIR Server	47
4.2	Building HAPI FHIR Environment	47
4.2.1	Prerequisites	47
4.3	Installing HAPI FHIR	50
4.4	Testing Hapi API with Postman	51
5	Solution Implementation	52
5.1	Project Structre	52
5.2	Create Patient Record	55
5.3	Create SNOMED-CT for Blood Pressure	58
5.4	Read SNOMED-CT	61
5.5	SNOMED Search	61
6	Discussion	64

7 Conclusion and Future Work	66
7.1 Conclusion	66
7.2 Future Work	67
Bibliography	67

List of Figures

- 1.1 Basic Design of Web APP 7

- 2.1 Levels of Electronic Health Records[39] 10
- 2.2 EHR Communication Standards[39] 11
- 2.3 Healthcare data interchange standards 12
- 2.4 HL7 Version 2 and 3 [13] 14
- 2.5 FHIR Patient Resource Figure [14] 15
- 2.6 Overview of FHIR [15] 16
- 2.7 Example of primitive type [16] 17
- 2.8 Example of Complex data type [16] 18
- 2.9 Example of Complex type JSON [16] 18
- 2.10 FHIR specification components [17] 19
- 2.11 FHIR Composition Framework [18] 20
- 2.12 FHIR profile for implementation 23
- 2.13 Extention Reference Example JSON [22] 24
- 2.14 Terminologies Relation [23] 26
- 2.15 UML diagram of Patient Resource [26] 28
- 2.16 UML diagram of Patient Resource [28] 29
- 2.17 Observation Profiling of blood pressure 30

2.18	UML diagram of Practitioner Resource [29]	31
2.19	Relation Between Resources	31
2.20	SNOMED Browser	34
2.21	SNOMED Design	35
2.22	SNOMED Code Example	36
2.23	SNOMED Versions [35]	37
2.24	Hapi Architecture [36]	37
3.1	System Architecture	40
3.2	Client Web Application Architecture	41
3.3	CliniFHIR Structure of FHIR message	42
3.4	Observation Created using Clinifhir	43
3.5	Design to create and search Resources	44
4.1	Index Page of Tomcat server	48
4.2	Adding Path Environment Variables	49
4.3	HAPI FHIR Server	50
4.4	Postman sending Post request to FHIR server	51
5.1	Project Structure	54
5.2	Create Patient Record	55
5.3	OnClick Function called Named SavePatient	55
5.4	SavePatient Method Contains the FHIR messages	56
5.5	HAPI FHIR server response	57
5.6	Our Application Response	57
5.7	FHIR resource populated	58

5.8	Observation Form	59
5.9	SNOMED-CT created in HAPI FHIR resource	60
5.10	HomePage of our Application	63

Chapter 1

Introduction

Proper exchange of medical information provides the patient, practitioners and medical researchers ability to ease and share securely medical information of the patients. Development of the medical systems and interoperability will help the patient in the medication process, reduce the wrong medication and risks. Standardization is important for interoperability in health information system in different aspects. For example, standard healthcare vocabulary, exchange of healthcare information standardized by Health Level Seven International (HL7), Snomed-CT for improving patient records , accessibility of information and secure the information [1].

In the last decade, growth in the field of IT is exponential and it provides a solution in every field of life. The field of information and communication (ICT) is the emerging field which supports the healthcare system i.e., health education and health surveillance. According to the Norwegian Directorate of eHealth (NDE), more development is needed in the systems used by the health care professional. It is reported by NDE that healthcare in Norway consists of different independent organizations which have their own priorities and different solutions [2].

A survey by the Transcend Insights in 2017 on the sharing of the clinical data shows that 64 percent of the patient use the medical devices to monitor their vital signs. Most of them think that this data could be useful for the practitioner and should be stored in their electronic records. The clinical data is stored in electronic patient records, which

is a digital version of the patient's paper record [3]. For example, when a practitioner gets access to the electronic patient record like demographics, allergies, medications, test results from the laboratory, previous treatments and other clinical data, It must be possible to understand the information regardless of which clinical institution accesses the patient record. Electronic patient records can have a positive impact on the quality of the treatment, patient safety and can provide better results in the health systems. To achieve these benefits, it is not enough to save the data electronically, but it requires the patient record to contain accurate and relevant data in a usable format that can be accessed.

Due to this reason, the exchange of clinical information is crucial which requires both information model and clinical terminologies to function together, i.e., FHIR (Fast Health Interoperability Resources) and SNOMED-CT(Systematized Nomenclature of Medicine-Clinical Terms). Over the past years, the standardization community has started to work for better solutions for clinical terminology binding. HL7 TermInfo and CIMI (Clinical Information Modelling Initiative) worth mentioning [3].

One of the important terms is vital signs which indicate the status of the body functions. It helps the practitioner to assess the health of the patient, which helps in diagnosing the disease and shows the progress of patient health. There are four primary types of vital signs: blood pressure, temperature, heart rate and respiratory rate. In our thesis, we are working with blood pressure using the FHIR as an information model with SNOMED-CT terminology. The sharing of the vital signs across the healthcare system provides decision support and help in the surveillance.

This work aims to implement the web application which provides the profiling of the vital signs and provide the search capabilities. FHIR is used as an information model and SNOMED-CT as terminology to represent vital sign information.

1.1 Background

With more usage of Electronic health record (EHR) in different healthcare organisations, increased the demand for semantic interoperability in clinical research. The lack of in-

Interoperability of EHR between different organisations makes it difficult to reuse the data later. Unfortunately, today we see that the health services in Norway and other countries consist of several systems that have problems interacting and sharing the information. This is partly because the systems lack a suitable data structure and a common standard for data exchange. Several systems today use their own service structures for sharing data and these are often difficult to obtain access to [4]. This prevents health care from giving patients the best possible treatment.

Interoperability is generally recognized as a key requirement for the success of the health-care information system [9].

Successful interoperability delivers economic value - the economic cost benefit of interoperability of healthcare information systems in the USA alone has been estimated at USD 77.8Bn dollar[9].

To provide such interoperability and instant exchange of patient information requires the use of standards. The primary role of standards is to define the data structure for the exchange of information from one health system to another. According to the Academic committee for health and architecture in Norway (NUFA) here are the list of standards and specification defined for the health care system in Norway.

OpenEHR: is a non-profit organisation facilitates the creating and sharing of health records, standards-based implementation, including decision support rules, query language and modelling of the health data. [5]

HL7 Version 3 Messaging: It is a Health Level Seven version 3 standard based on Reference Information Model (RIM) and messages are exchanged in XML (Extensible Markup Language) syntax with focus on information semantics [6].

HL7 CDA R2: is a clinical document standard that provides semantic and structure to the clinical document (procedure report, summary report). It includes complete information including images, texts, sounds and multimedia content [7].

IHE XDS.b: Cross-Enterprise Document Sharing (XDS) is an integration system which

provides the standards to manage and share the documents with no limitation on the type of document.

Link Data: it provides the connectivity to the data by using the W3C (World Wide Web Consortium) standards, HTTP based URI is used, SPARQL and (Resource Description Framework)RDF standards provide the functionality and link to the other URI names.

HL7 FHIR stands for Fast Health Interoperability Resources is a set of standards defined by the HL7, which provide the communication/exchange of health information between different system. FHIR gathers the best features of HL7 v2, v3 and CDA. FHIR solution is based on the resources which provide wide medical information. FHIR uses the latest web-standards for the developers and implementors [8].

FHIR is a successor of previous standards, one of the advantages of FHIR over previous version that the support of RESTful architecture. According to the assessment made by Norwegian Directorate of Health that FHIR is the only international standard which is the recommended solution for the REST-based architecture [2]. This assessment is made by considering the different model of interaction i.e., document sharing, message exchange, text exchange etc. and FHIR is the only standard that supports all the interaction model. Moreover, the base resources in the FHIR can be adjusted for the local development and provide interoperability as well [8].

SNOMED-CT: The systematised nomenclature of medicine SNOMED-CT began in (National Institute of Health) NIH. Interestingly the original objective in the 70s was machine coding of a pathologist free text dictated a note. It was always in ontology representing relationships among its concepts. Today it is expanded to all the medicine and maintained by the International Health Terminology Standard Development Organization (IHTSDO). SNOMED is huge even the SNOMED-CT has 311000 concepts and represent 1.3 million relationships among them. Concepts are the basic component of the SNOMED-CT and have clinical meaning. They are identified by the unique 9-digit numeric concept ID and unique human-readable fully specified name [10].

Our goal of this thesis is to make a web application HL7 FHIR based architecture, which provide the ability to send, retrieve and search the patient vital signs including SNOMED. Thereby, if sending a query the information will become available using Rest-based services and represented in FHIR format.

1.2 Motivation

The motivation of this master thesis is to reduce the risk in patients treatment that may happen due to the lacking of interoperability between different medical system. Our idea is to represent the patient vital signs using SNOMED-CT terminology which improve the accessibility and support decision making.

1.3 Research Questions

Q 1. How to carry the SNOMED-CT using the FHIR protocol.

Q 2. How to query the SNOMED-CT coded vital signs from FHIR enabled server using the FHIR protocol.

Q 3. How a web-based client exchange vital signs data with FHIR enabled server.

1.4 Proposed Solution

The exchange of information at the level of semantic interoperability needs the information model and clinical terminologies work together. Applying the HL7 FHIR standard with SNOMED-CT will resolve the issue of interoperability. The assessment made by Norwegian-e-Health directorate on international standards, HL7 FHIR is recommended. According to the HL7, FHIR interaction model can be used in four ways: document exchange, information exchange, document sharing and information sharing [14]. In this thesis, we adopted HL7 FHIR specification to build a new FHIR data model and clinical terminology solution.

There are different FHIR based solutions for terminologies that are in developing or deployment phase. Apelon is clinical informatics company which provides solution based on clinical terminologies. It provides the single point of contact for different standards, mapping of terminologies and a software tool to manage the system [42].

Intelligent Medical Object (IMO) is a clinical interface terminology which facilitates to capture the clinical information. More than 45,0000 physicians and over 4000 hospitals and medical centers use IMO for a better workflow for electronic health records. It provides the software to manage the clinical information and terminologies [43].

1.4.1 Developing a Restful services

For the restful services we are using the HAPI FHIR server which is the JAVA library build according to the specification of FHIR [11]. Our implementation will be developed in the JavaScript programming language and we will use the Vue.JS framework to build the web application.

1.4.2 Interaction of Web application with Restful services

We will develop web based front-end interface to access the SNOMED-CT terminologies and patient data from FHIR resources. In the below figure shows the design of the web application. We are developing the pilot application but we have kept in the scalability of the project in future.

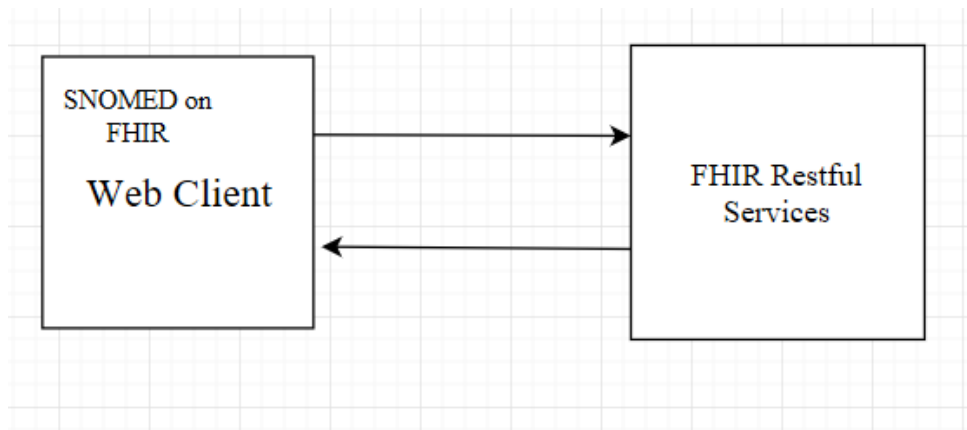


Figure 1.1: Basic Design of Web APP

1.5 Report Structure

This report is divided into five chapters: Chapter one represents the introduction of the topic. Chapter 2 consists of a theoretical background and related work. It contains the review of the electronic health record (EH.R), literature review, the evolution of Health level seven international (HL7), discussion about different standards of interoperability in the e-health industry. Fast health interoperability resources FHIR, discussion about clinical terminology, the importance of clinical terminology, SNOMED-CT origin and their structure, how we can use the SNOMED-CT with FHIR. HL7 FHIR release 4, HAPI FHIR server and information about the development tools i.e., Tomcat, JavaScript, JSON, Apache Maven, Vue.js framework and web application development.

Chapter three includes the system design of our application and explain the procedure to query the SNOMED-CT terminologies.

Chapter four discuss the solution deployment pre-requisites, in which it include the procedure to install the HAPI FHIR server and testing of HAPI API. Chapter five includes details about the implementation of our solution. We explain step by step the procedure to build our web application. Starting from the prerequisites, designing of the system and implementation of the solution. Finally the validation of our solution.

In chapter six contains the discussion. The outcome of our solution and discussion on

research questions We have described some challenges we faced during the implementation process.s Chapter seven includes the conclusion and future work.

Chapter 2

Theoretical Background and Related Work

2.1 Electronic Health Record EHR

The idea of the electronic health (EHR) record is to store and share the patient information digitally, evolved through times and make it easier for the practitioners and hospitals to store the information easily. EHR has solved the many issues including efficiency, communication, reduce the medical error, bring down the health cost and improved the workflow to provide better and effective treatment.

When this concept was introduced many organization and institutes implemented this solution, as a result, different classification, categorization and customization was made for patient records. For example one of the earliest methods is Automated Medical Records (AMR) in which they used to scan the patient document to store it digitally. Although it helped to decrease the use of paper but it is not helpful when it comes to decision making. In comparison with Electronic Medical Record (EMR) it is not complicated. In EMR all the lab result, diagnosis reports, documents, digital images are stored in it. We can say that EMR supports the decision making which got more appreciation from the health industry.

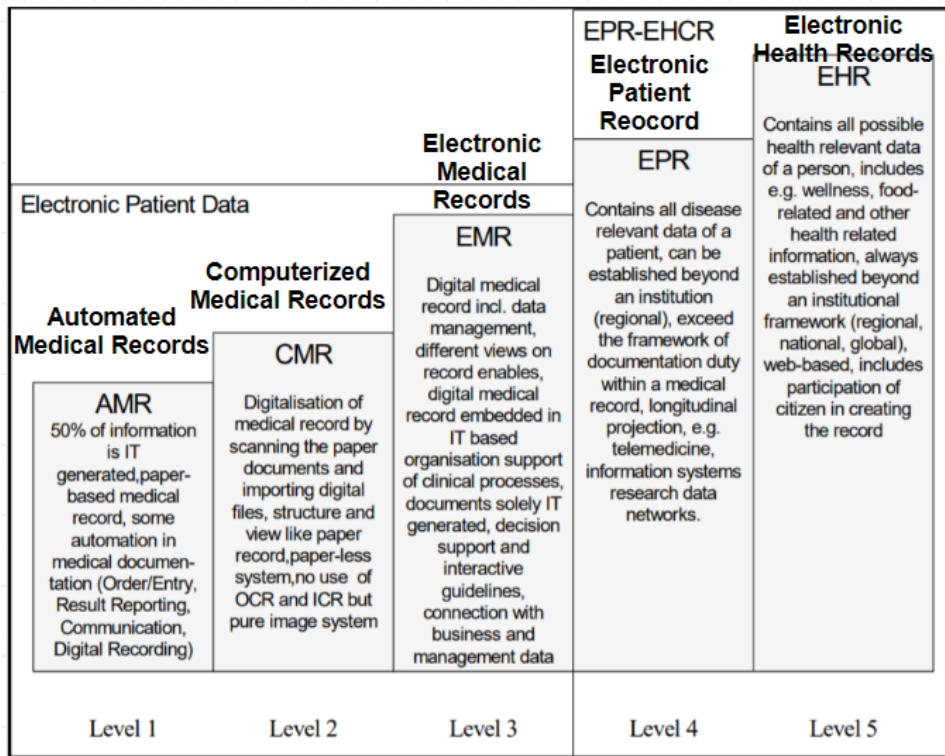


Figure 2.1: Levels of Electronic Health Records[39]

According to the Medical Records Institute the electronic health records are classified into 5 different levels illustrated in the figure below.

If we look from a development point of view, among health care system medical information is exchanged from one health system to another requires the complicated communication standards. It is a challenging task to define such standards shown in the figure below. The figure illustrates the number of standards that are used for different purposes. There is a one EHR system which used the IEEE standards for communication of medical devices and HL7 for clinical reasons. Hence exchange of information becomes more complex.

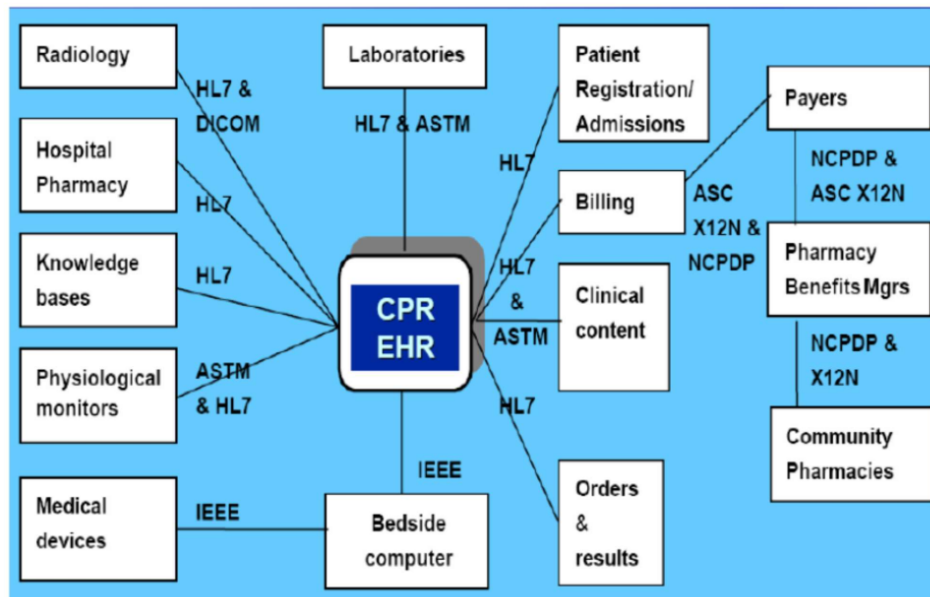


Figure 2.2: EHR Communication Standards[39]

2.2 Health Level Seven International (HL7)

The development of the standards for health information system needs to fulfill requirements that support the system for a long time, it includes the technical committee of experts, discussions on pros and cons, issuing drafts and voting. And if the draft is accepted it becomes the standard for practical use i.e., HL7. HL7 is one of the most used standard in the health information industry. The number 7 in HL7 represents the seventh layer (application layer) of the Open System Interconnection (OSI) model. The application layer is responsible for interpreting the send and receive responses.

The HL7 was founded in 1987, a non-profit organization. It has partners from all the continents with members more than 55 countries. Who represents the number of health information systems manufacturers, consultants and government agencies [12]. HL7 provides the specification to exchange the administrative and clinical data among different health information systems. Furthermore, it provides the related standards and framework for the integration, exchange, retrieval and sharing of health information that supports the clinical practice and management [12]. The most used services of HL7 are messaging standard which facilitates the different healthcare application to exchange the administrative

and clinical data.

Here is the figure of standards for the interchange of healthcare data

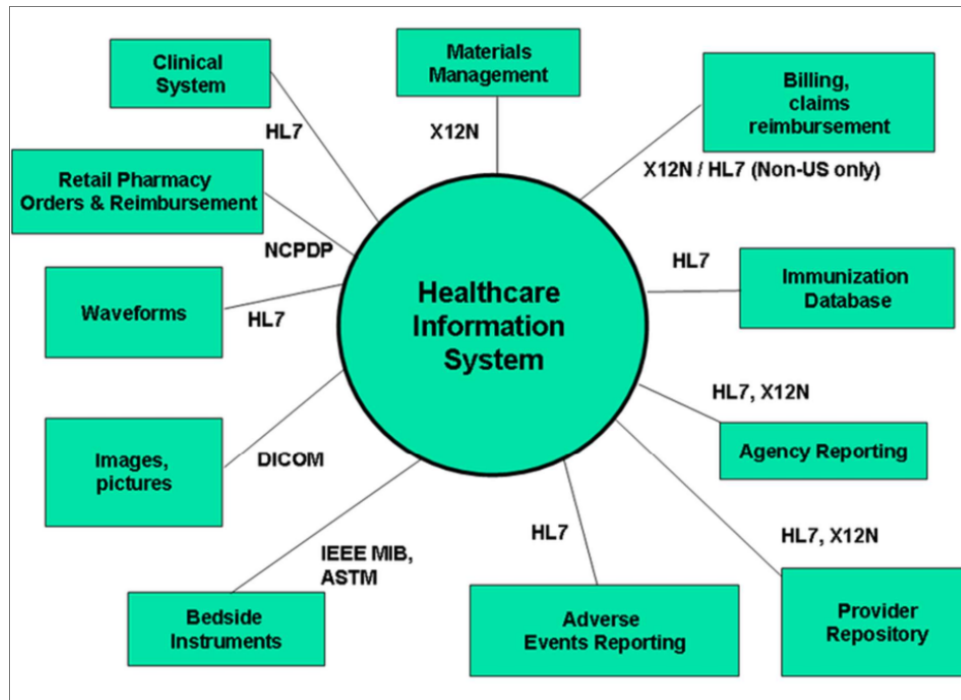


Figure 2.3: Healthcare data interchange standards

Following are the development authorities that have an agreement with HL7.

- SNOMED-CT: provide the global recognized clinical terminology for health information system.
- Center for Disease Control Prevention.
- DICOM: facilitates in exchange of clinical images
- ADA: used to process data for data services
- ANIA: used to process the data for nursing services
- ASTM: used to exchange messages for observation performed in clinics

2.2.1 Evolution of HL7

As discussed earlier, HL7 is the framework for clinical standards for exchange of medical data. When the first version of HL7 came out, it represented the structure of interfaces

between systems and was required to develop the interfaces for every system to exchange the data. When HL7 version 2 released, the goal is to provide the standard framework for the exchange of clinical data. Everything in version 2 is text-based, the information is separated by the text character pipe . To read the information the user is required to know the order and need to learn different glossaries to sort out the meaning. The main reason version 2 became popular in the market because of the predefined approach for interface and left some room for the health care provider to customize their requirement.

After V2, the next version was released that came with the feature of developing web technologies and extensible Mark-up language (XML). More flexibility is introduced for human readability. Clinical Document Architecture (CDA) is introduced along with version 3. While other versions are focused on the exchange of information, CDA is focused documentation of the patient (example: patient diagnosis report document). Design to provide more human readability when it came to the patient. Figure below shows the difference between version 2 and 3.

```

MSH|^~\&|RAD|HOSP||200910052215||ORU|RMS|P|2.3|
PID|1||0000123456^M10|000000000000|BLUTH^MICHEAL^^^|000123456^9^M10^^
PV1|1|0|111111|E|HOSP||200910052054|
OBR||H79058737-2-1^GPHI|2594572|11780^KNEE 3 VIEWS^RAD||200910052148||PSC||||^^^L||||200910052215||F|1^N
OBX|1|XCN|11780&SDR^KNEE 3 VIEWS^RAD||059980^MCCOY^BONES^|
OBX|2|TX|11780&GDT^KNEE 3 VIEWS^RAD||Left knee and left lower extremity: 10/5/2009 Indication: Tenderness and pain.
OBX|3|TX|11780&GDT^KNEE 3 VIEWS^RAD||~Transcribed by - PSC~Transcription Date - 200910052215||F|
OBX|4|TX|11780&IMP^KNEE 3 VIEWS^RAD||Normal knee radiographs ||F|

```



Figure 2.4: HL7 Version 2 and 3 [13]

2.3 Fast Healthcare Interoperability Resources (FHIR)

Fast Healthcare Interoperability Resources is the latest standard framework created by HL7. FHIR has combined all the features of HL7's previous standards, i.e., V2, V3, and CDA. FHIR is a robust framework with flexible implementation for web technologies. FHIR is built of modular components called Resources [14]. Only the resources will send between the players. It is easy to implement these resources into a working system that can solve clinical and administrative problems at low cost compared to existing alternatives. FHIR can be used in a variety of context, e.g., desktop application, mobile apps. HER based sharing, cloud communication and server communication [12].

As FHIR is more focused on the implementation, there is a number of libraries available for the development. Furthermore, FHIR supports several web technologies, i.e., JSON,

XML, OAuth, HTTP that make flexible and scalable the FHIR system. One of the main challenges in the development of the healthcare system is the continuous demand for adding new field and option which increases the total cost and complexity as well.

”FHIR solves this challenge by defining a simple framework for extending and adapting the existing resources. All systems, no matter how they are developed, can easily read these extensions and extension definitions can be retrieved using the same framework as retrieving other resources” [14].



Figure 2.5: FHIR Patient Resource Figure [14]

2.3.1 HL7 FHIR Release 4

The version we are using in our implementation is the new FHIR release 4.0.0 (released in 27-Dec-2018). There are various reason for choosing FHIR for the project i.e., Based on the web standards like JSON and HTTP, documentation available, the availability of

implementation libraries and the support RESTful architecture [15]. The structure of FHIR consists of levels from 1 to 5, where each level has a set of modules. A module consists of relevant resources. Below figure illustrates the structure of FHIR.

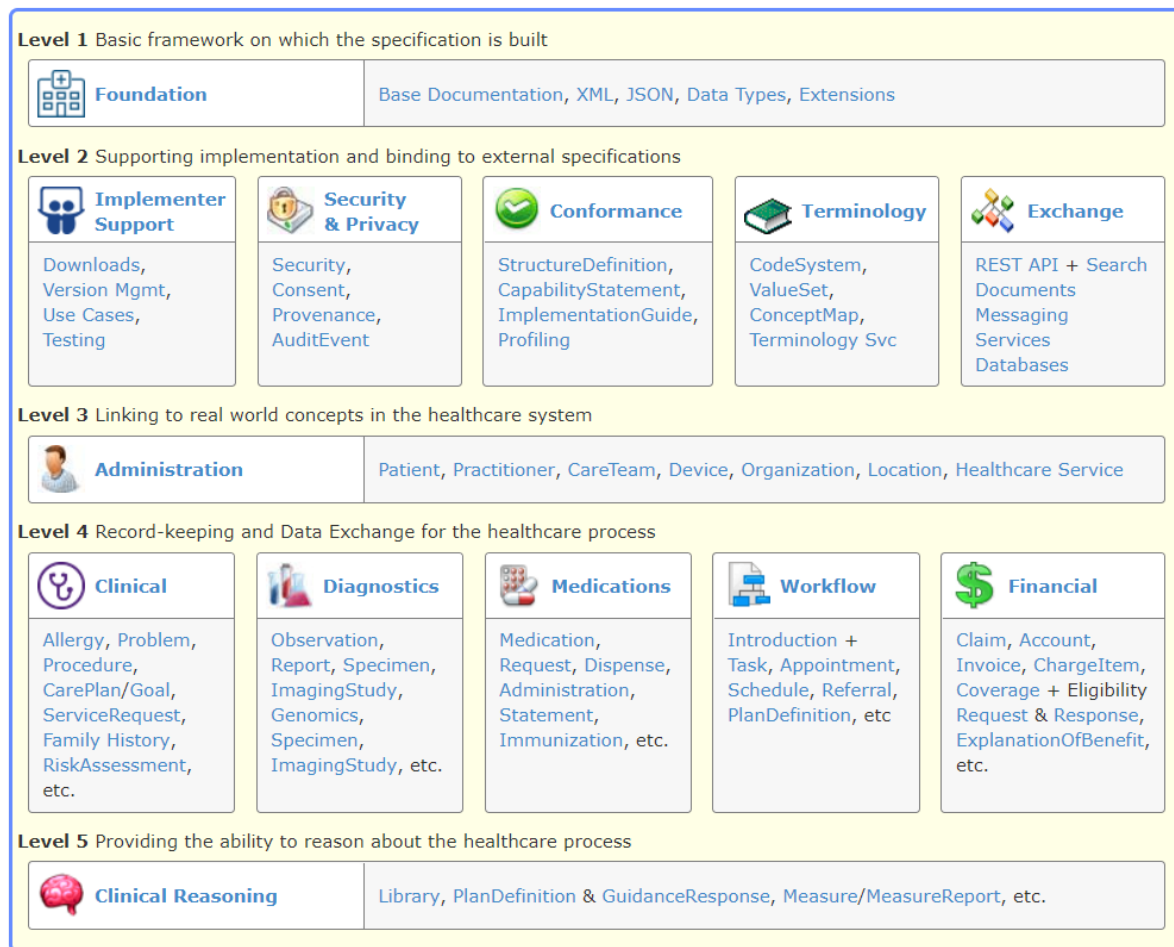


Figure 2.6: Overview of FHIR [15]

FHIR is a implementation oriented standard which emphasis on the integrity of the information. FHIR is a flexible standard can be used as standalone or used with other standards as well. Resources are the main building block which is used to exchange the information. Resource consists of different data types, i.e., complex and primitive data types. Complex data can hold the child element, but primitive data type does not have the child [16]. Resources have a field called metadata which contains the human-readable part. The modeling of health information is accomplished by using FHIR resources to define the relationships between different objects of most common cases with a capability if using extensions for specific content. Example of primitive type in JSON count: 2.

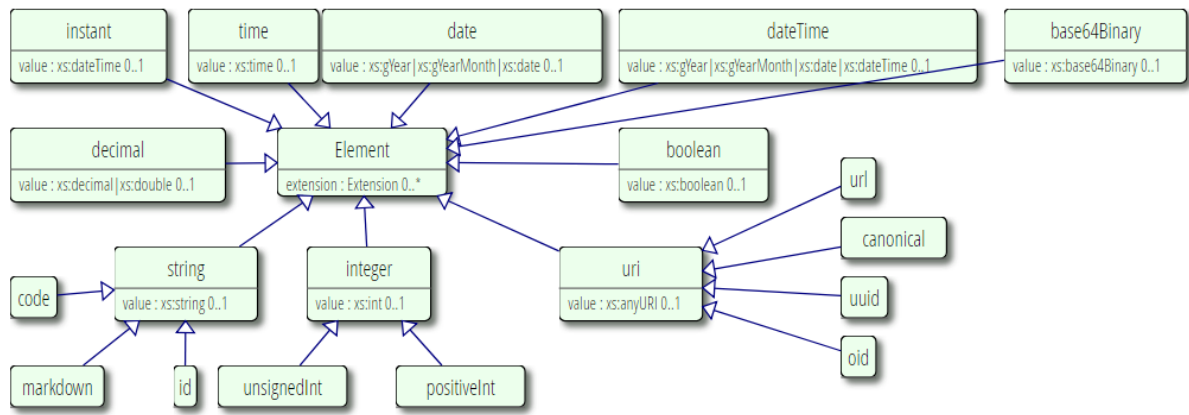


Figure 2.7: Example of primitive type [16]

The Example of complex data type and human readable in JSON is illustrated in the figure below. To model the healthcare information FHIR resources are used to define the relation, sometimes extensions are used to add the new capabilities for specific content.

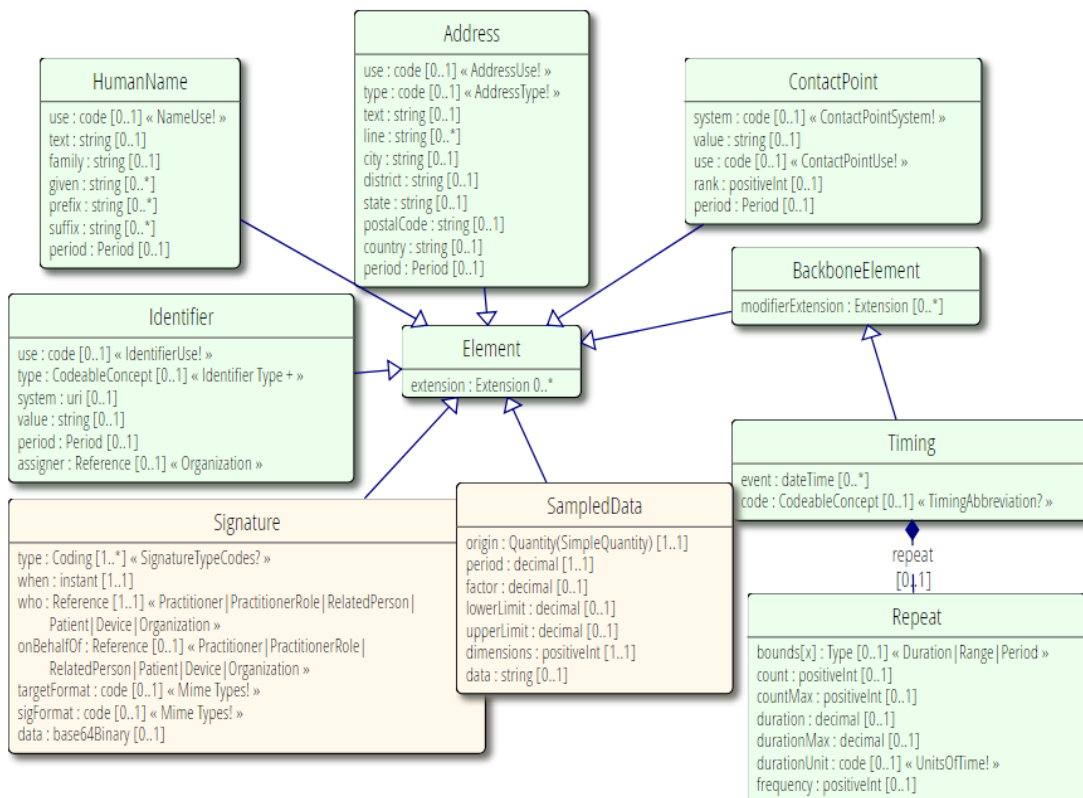


Figure 2.8: Example of Complex data type [16]

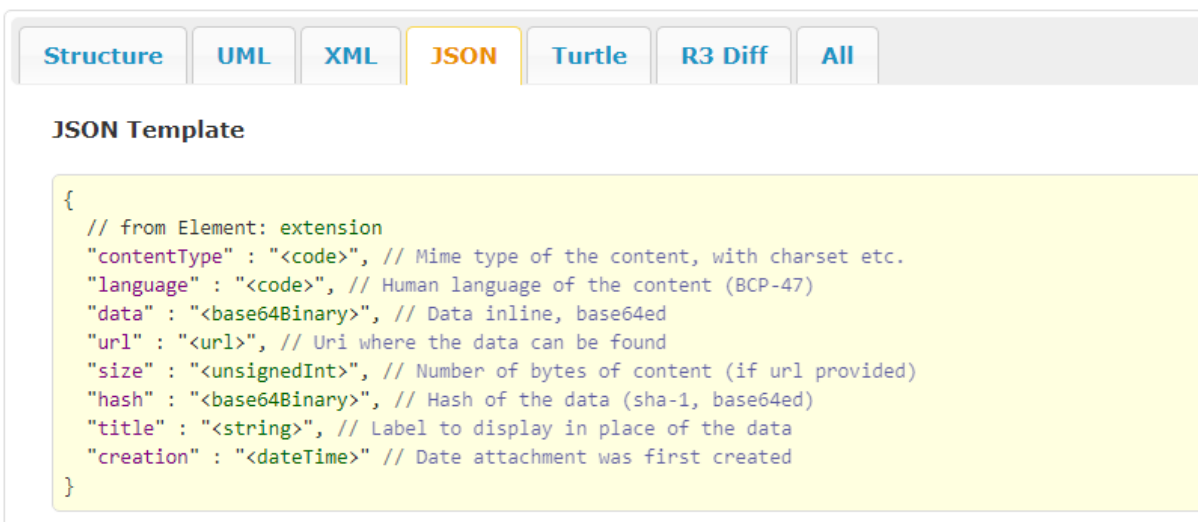


Figure 2.9: Example of Complex type JSON [16]

2.3.2 Structure of FHIR

FHIR is divided into four components.

1. Conformance (provide flexibility in different health data system).
2. Information Model (related to FHIR resources).
3. Terminology (provide the support to different terminology like SNOMED).
4. Usage (Interoperability)

Relation between these components is illustrated in the figure below:

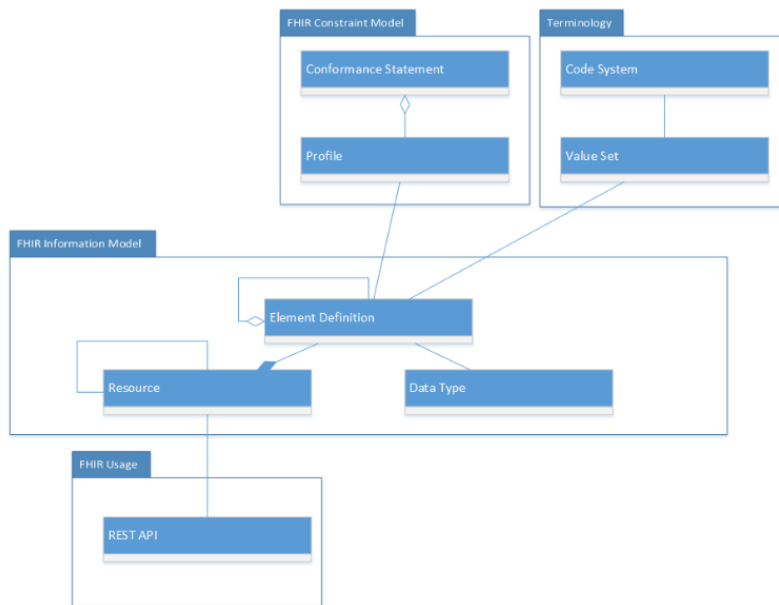


Figure 2.10: FHIR specification components [17]

2.3.3 FHIR Composition Framework

To create the model for health data, FHIR represents healthcare information as a domain which consists of small sub-domains. As resources are the basic building blocks so each sub-domain has related resources to it. To organize the information FHIR use the FHIR composition framework. It supports the identification, navigation, grouping of the related resources and provides the functionality to add new resources. The figure below shows the FHIR composition framework.

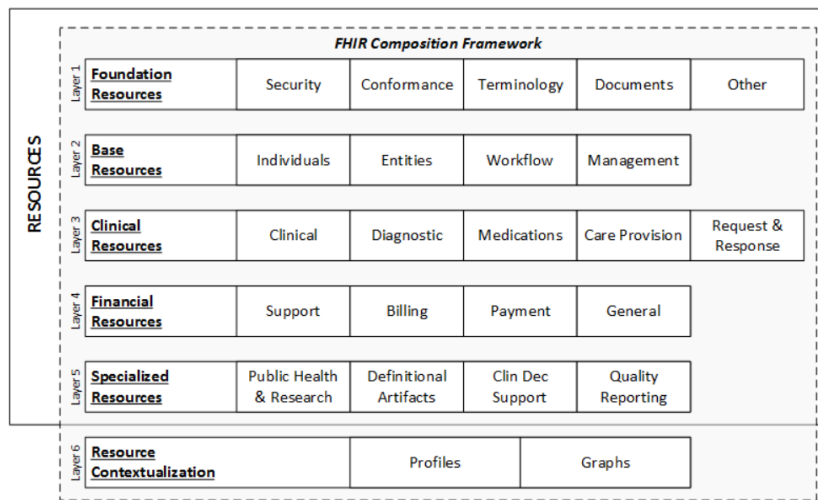


Figure 2.11: FHIR Composition Framework [18]

2.3.4 The Four Paradigm

FHIR supports four paradigms to transfer the information: RESTful API, Messaging, Services and Document. In our work, we are focus on the RESTful approach since it is mostly used in the implemented application prototype that has been developed during this work.

2.3.5 FHIR RESTful API

The Application Program Interface (API) defines the set of operation that can be performed on the FHIR resources following the condition that the server should provide the

capability to support the operation. REST stands for the Representational State Transfer. The word Representational point out the request of a specific resource: any instance of the source, for example, the user can get the patient information by sending the relevant URI. The word STATE represent the information about the state of the information the user has requested. This information can be represented in the XML, JSON or CSV format. The HTTP verbs (GET, POST) provide the interaction between the client and the server. Consequently, the RESTful approach provides the scalability that is very important for the healthcare information system. FHIR API provides the number of interaction operation with the resources. These operations are read,delete, update, create and search. One of the examples that retrieve the information from the patient resource is GET.

Following is the syntax that defines the interaction with FHIR resources: **VERB** [**base**]/[**type**]/[**id**]

[**base**] represents the Service Base URL where all the resources defined by the interface.

e.g., `http://fhir.anyserver.org/fhir/patient/1`

Verb: GET

Base URL: `fhir.anyserver.org/fhir`

Resource type: patient

Id value: 1

The interaction offered by the FHIR API are following [18].

Create interaction:	POST	<code>https://fhir.anyserver.org/path/{resourceType}</code>
Read interaction:	GET	<code>https://fhir.anyserver.org/path/{resourceType}/{id}</code>
Update interaction:	PUT	<code>https://fhir.anyserver.org/path/{resourceType}/{id}</code>
Delete interaction:	DELETE	<code>https://fhir.anyserver.org/path/{resourceType}/{id}</code>
Search interaction:	GET	<code>https://fhir.anyserver.org/path/{resourceType}?search parameters</code>

2.4 FHIR Concepts Need to Understand

In order to implement our concepts, there number of FHIR specifications which are required to understand. As we have mentioned the main building block is resources in FHIR standard that is used to exchange and store healthcare information. According

to the FHIR specifications “Every resource has its own unique URL by which it can be accessed, identifies itself as one of the types of resources defined in this specification, contains a set of structured data items as described by the definition of the resource type, and has an identified version that changes if the contents of the resource change[18].”

The logical identity to the resource is given by the server which is handling the request: the example of the logical identity of the patient resource is `http://fhir-server/Patient/321`. One of the features of the FHIR is the support for the previous version. Whenever there is a change in the structure of any resource, e.g., Patient, the previous version can be accessed by using the URL.

“`http://fhir.anyserver.org/history/1`” and the new version can be accessed by using the URL `http://fhir.anyserver.org/history/2`

In FHIR all the resources have following similar features [15].

1. Unique URL which identifies the resources.
2. Metadata
3. A human-readable XHTML summary
4. A set of defined data element (can be different for every type resources)
5. Extensibility framework to support variation in healthcare.

Sometimes the Resources and Resource instance are considered interchangeable which can lead to misinterpretation. A Resource term is used when we mention to the resource as a data type and Resource Instance used during the instance of the resource. As stated by the FHIR specification a resource instance contains the id, metadata, resource id, human-readable XHTML, data, extensions, and identifier.

2.4.1 FHIR Profiling

As the FHIR specification explains the set of base resources, API and framework that are used in different context in the healthcare system. However, there should be some rules which resource must follow, which technologies could be used [19].

For the observation of vital signs which has various levels can be represented by using the FHIR profiling so that local requirement can be handled [19]. We have gathered the set

of customized implementation guide and include the profile about the changes made to resources. An example of an implementation guide is shown in the figure

```
{
  "ImplementationExample": {
    "url": "http://uia.health.no/vitalsigns",
    "name": "vital-sign implementation for FHIR",
    "status": "draft",
    "date": "2019-04-21T18:25:05.456781+02:00",
    "description": "ImplementationExample",
    "fhirversion": "4.0.0",
    "package": {
      "name": "vital-sign-observation",
      "description": "Profile for vital signs",
      "sourceReference": "http://uia.health.no/Observations"
    }
  }
}
```

Figure 2.12: FHIR profile for implementation

According to the FHIR specification vital signs profile draft, vital signs must conform to these points [26].

- Every vital sign must be marked the common terminology code e.g., LIONIC. In addition, you can assign whatever code to it like SNOMED-CT.
- There must be value of vital sign or need reason in case of absence.
- Diastolic/Systolic must be represented by using components.
- Must have the subject for observation.
- UCUM (The Unified Code for Units of Measure) unit must be included

2.4.2 FHIR Extensions

“The paper forms (Resources) in FHIR are somewhat generic. They have to be usable in different countries and by different types of clinicians in different contexts (human care,

veterinary care, public health, research, etc.). Recognizing that a one size fits all approach is not appropriate in the healthcare space, FHIR provides the ability to adjust the forms (Resources) to be able to handle the needs of different implementation spaces by defining “extensions” as well as enforcing constraints [20].

It is important to keep everything in accordance when adding the new elements to the resources for a specific used case i.e., adding the extensions. It looks straightforward but there are some things which must be kept in mind.

The Extension Definition: The StructureDefinition resource is used to hold the definition of extensions.

ValueSet: is a resource which explains the possible value for coded element. Usually refers to the one or more CodeSystem which describe the actual concept.

CodeSystem: It consists of the list of concepts that has a code, represent the external terminology i.e., SNOMED-CT.

STSTEM: Technically, it is a URL. It is the child element found in the coded elements and Identifiers that links the element to a set of permitted values.

NamingSpace: Defines a specific code system or identifier system, so that it can be noted in a registry for other systems to find and understand an identifier [21].

The FHIR profile achieve the customization by adding the structural definition called Extensions. These extensions can have any FHIR datatype e.g., the reference to another resource.

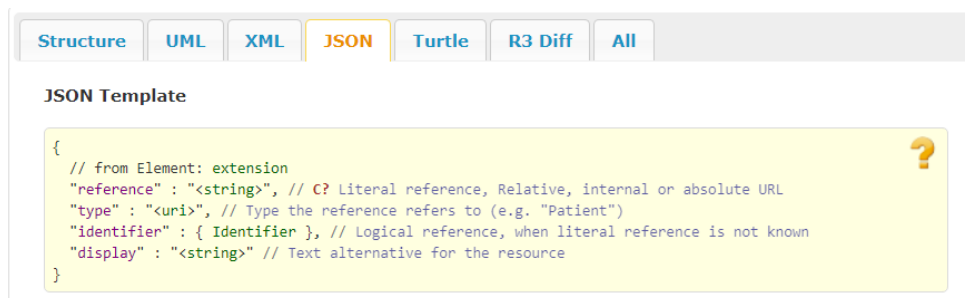


Figure 2.13: Extention Reference Example JSON [22]

2.4.3 FHIR Terminology Module

The Terminology Module provides an overview and guide to the FHIR resources, operations, coded data types and externally-defined standard and FHIR-defined terminologies that are used for representing and communicating coded, structured data in the FHIR core specification and profiles. Collectively, these capabilities are used to provide the terminology service functionality required for supporting the use of coded data in FHIR resources throughout the specification as described in the other modules. [23]. The figure below shows us the relation between resources and terminology.

The terminology module consists of following resources.

- CodeSystem
- ValueSet
- NamingSystem
- ConceptMap
- TerminologyCapabilities

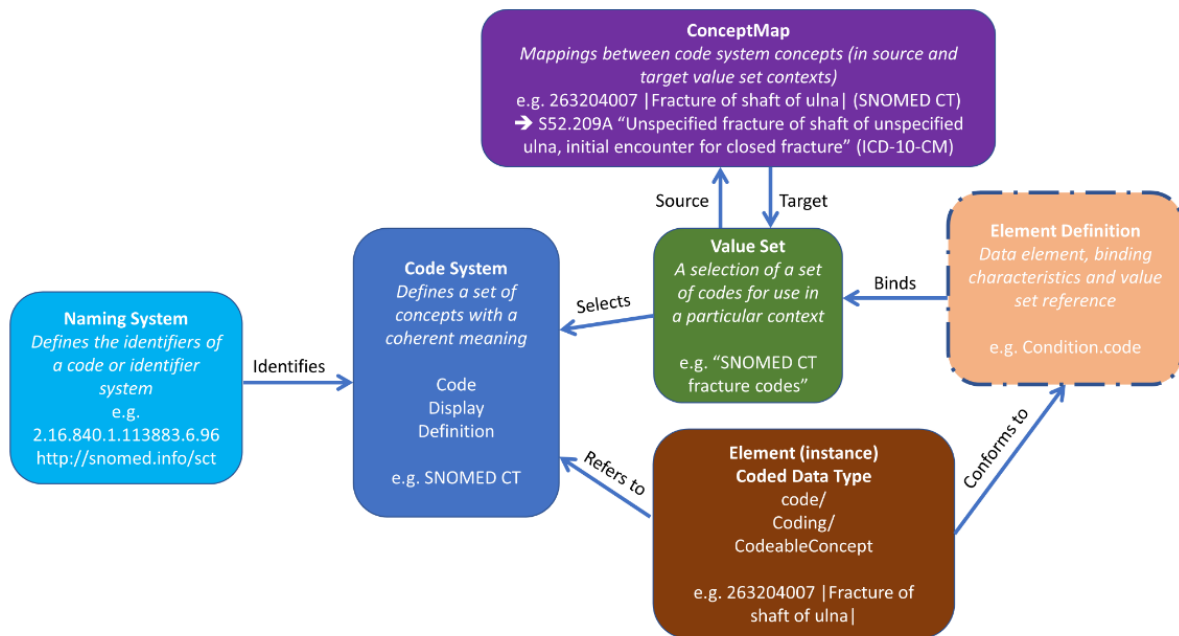


Figure 2.14: Terminologies Relation [23]

2.5 Focus on FHIR Resources

We are going to focus on the different FHIR resources which we are going to use in this thesis and lies under the following modules [24].

1. Administration
2. Diagnostic
3. Terminology
4. Exchange

The main objective is to send the SNOMED-CT data using the FHIR as an information model. In this thesis, a use case describes the procedure to send the SNOMED-CT using FHIR. We need to make the profiling of the patient in order to achieve our goal. We will create a vital sign profile, and choose the following resources according to our requirement.

1. Patient

2. Observation
3. Practitioner

2.5.1 Patient Resource

According to the HL7 FHIR standard documentation the patient resource “ Demographics and other administrative information about an individual or animal receiving care or other health-related services [26].

It is important to know the scope of the patient resource. The data included in the resources need to be within the range of the resource covers. Patient Resource covers a wide range of health activities including: [26]

1. Curative activates
2. Psychiatric care
3. Nursing and assisted living
4. Dietary Services
5. Tracking of personal health and exercise data
6. Social Services
7. Pregnancy Care

The information about the patient is published by the health organization that provides the care for the patient. It provide the capabilities to refernce.For example, if the patient receives the care from the multiple health organization in this particular case multiple patient resources exists and referenced with each other. The (Unified Modeling Language)UML diagram illustrates the patient resource structure.

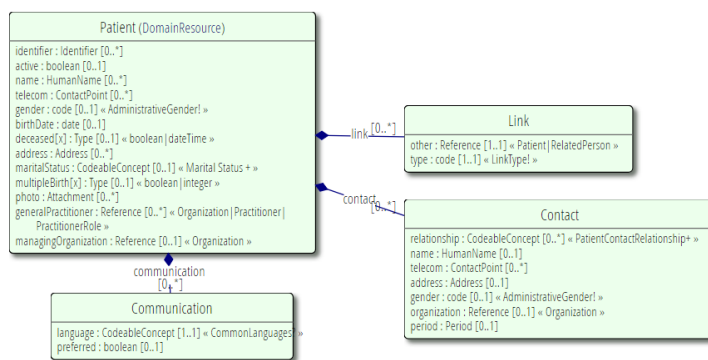


Figure 2.15: UML diagram of Patient Resource [26]

2.5.2 The Observation Resource

Observations play a significant role in the healthcare used to monitor progress, support diagnosis determines baselines and even capture demographic characteristics [28]. Usually, observations are value/name and some assertions with metadata. The uses of the observation resource include the following:

1. Vital signs such as temperature, blood pressure, body weight, etc.
2. Laboratory data like a blood glucose.
3. Device measurements.
4. Clinical assessment tools such as the Glasgow Coma Score.
5. Imaging result like bone density.
6. Measure the social history like family support.

An FHIR observation has only two mandatory fields; status and code. Status is a FHIR code describing the status of the observation and may have one of these 8 codes; registered, final, amended, corrected, canceled, entered-in-error and unknown. The code field describes what type of observation it is and can be.

In observation, reference is used to refer the patient and person who performed the observation is represented by performer. The stored on the separate resources, because FHIR provide the separate resources to store every information. In observation resource “field” that is not mandatory, however useful to include in sorting and classifying observations,

is the category where one can specify what type of observation it is. This field may have one of these tags; social-history, vital signs, imaging, laboratory, procedure, survey, exam and therapy. A simplified model of the structure of the FHIR Observation is illustrated in the figure below.

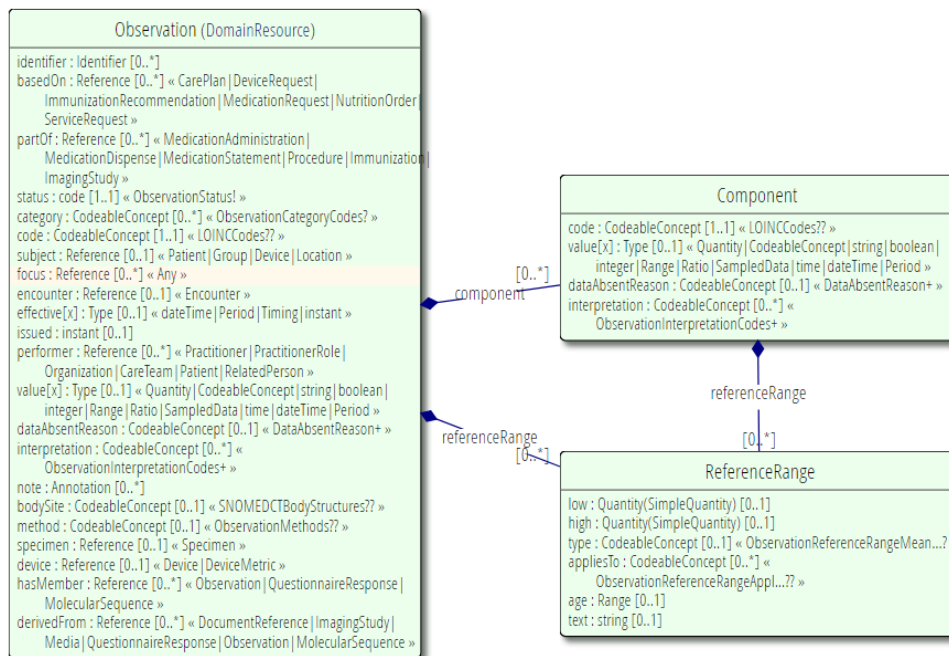


Figure 2.16: UML diagram of Patient Resource [28]

The actual observation can either be stored in a “value” field or by means of its own components of the ”Component” type. How the information is represented depends on which observation it has. If the observation has only a single value, such as the observation of body weight, this information can be stored using the “value” field and data type Quantity. Not all observations have only one value and then one must create ones components for each value in the observation. An example for this is blood pressure, which has two values for measurement; systolic and diastolic. A component consists of a corresponding “value” and “code”, which is used to describe the measuring values.

To represent SNOMED-CT in FHIR resource, we have created a profile of vital sign, which contains the blood-pressure as vital sign and SNOMED-CT as a terminology. Below figure illustrates how this profiling is performed.

```

{
  "resourceType": "Observation",
  "subject":{
    "reference": "Patient/example"
  },
  "code":{
    "coding": [{
      "system": "http://snomed.info/sct",
      "code": "163020007",
      "display": "Blood pressure reading (finding)"
    }]
  },
  "status":"final",
  "component":[
    {
      "code":{
        "coding": [
          {
            "system": "http://snomed.info/sct",
            "code": "163030003",
            "display": "Systolic Bloodpressure (finding)"
          },
        ] },
      "valueQuantity": {
        "value": 120,
        "unit":"mm[Hg]"
      },
      {
        "code":{
          "coding": [ {
            "system": "http://snomed.info/sct",
            "code": "163030003",
            "display": "Diastolic Bloodpressure (finding)"
          },
        ]
      },
      "valueQuantity": {
        "value": 95,
        "unit":"mm[Hg]"
      }
    }
  ]
}

```

Figure 2.17: Observation Profiling of blood pressure

2.5.3 Practitioner Resource

The person who is involved in the provisioning of healthcare. It includes physicians, dentist, nurses etc. There is the possibility to have more than one resource for each practitioner if a practitioner has multiple roles in the organization. Following the patient resource if all the information is not covered by the practitioner resource extensions can be added.

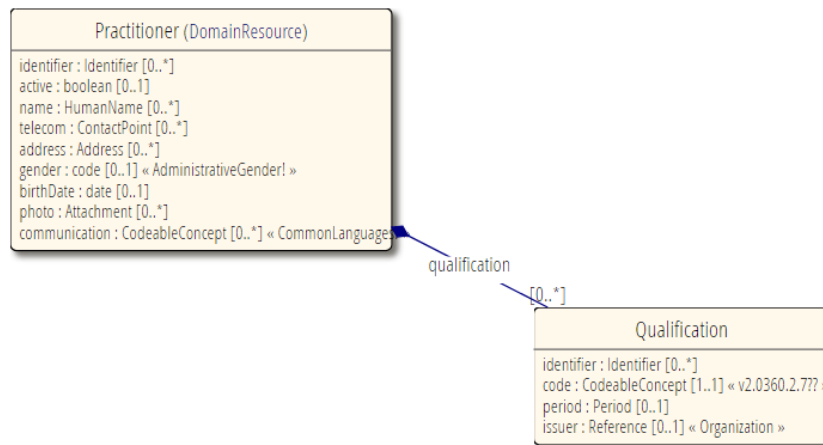


Figure 2.18: UML diagram of Practitioner Resource [29]

2.5.4 Relationship between Resources

In above mentioned resources the relationship between them can be expressed in terms of reference, for example the Observation resource has a reference to a resource called Patient. The following figure shows the potential references among them.

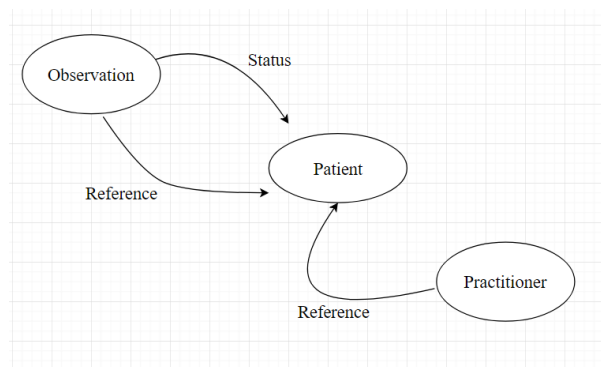


Figure 2.19: Relation Between Resources

2.6 Importance of Clinical Terminologies

Clinical terminologies are considered a key factor to enhance the communication of clinical data that increase the accessibility of relevant information. Medical terminology is important for many reasons: it provides the standard language for communication to healthcare providers and efficiency. It empowers the healthcare professional to diagnose better and helps to make decisions. The representation of medical information has become more specific over time. The clinical Terminology is defined as a set of concepts and relationships that provide a common reference point for comparison and aggregation of data about the entire health care process, recorded by multiple different individuals, system or institutions [31]. A clinical concept means a recorded value i.e., blood pressure value and it designed to enable the recording of accurate clinical information in an Electronic Patient Record (EPR). The information could easily retrieve from the record and its human readable or computer readable so it can be used for decision support

2.6.1 SNOMED-CT

As we are using the SNOMED terminologies to in our thesis it is important to know about terminology and their uses, advantages and how they can help in decision making. SNOMED-CT stands for Systematized Nomenclature of Medicine – Clinical Terms which is the most comprehensive multilingual clinical terminology available for health care. It is used to represent the electronic health record that facilitates the clinical documentation in retrieving, reporting and analysis of clinical data. It is maintained and distributed by the SNOMED international which is the non-profit global organization.

Why SNOMED? Suppose a patient has some chronic disease and taking medication. The practitioner wants to know the number of patients or side effects of the medication with the same disease. The system without SNOMED is not able to give the practitioner to do such functionality. Because every system is different and various health information system uses the different term to represent health data. If SNOMED clinical term is used

in each health organization then the practitioner can perform a search which helps in decision support for both practitioner and patient

2.6.2 SNOMED in Norway

Norway became a member of SNOMED International in 2017. In the same year, the Directorate for eHealth started a 3-year exploration period for how terminology should be introduced in Norway. On this basis, the Directorate for eHealth has decided that Norway should continue its membership and that SNOMED CT should be used where there is a need for standardized health terminology. The decision was endorsed in the National eHealth Board in June 2018. How SNOMED CT is to be introduced is still under investigation [31].

2.6.3 Origins of SNOMED

In 1999 the College of American Pathologists (CAP) and National Health Service (NHS) agreed to merge reference terminology (SNOMED-RT) with NHS Read Codes Version 3 also known as Clinical term Version 3 (CTV3). As a result, single joint clinical terminology came into existence called SNOMED-CT with a first version released in 2002. Later in 2007 International Health Terminology Standards Development Organization (IHTSDO) acquired all the rights to SNOMED [30].

SNOMED is big. The number of concepts, relationship and description increases with every release. It contains more than 300,000 active concepts and has 1 million description and more than 1.4 million relationships. There is no specific documentation of SNOMED concepts it can be accessed by using the specific software called SNOMED browser. The figure below shows the user interface of SNOMED browser.

SNOMED is huge it cannot be used manually but importantly it works differently compare to earlier code schemes it defines the relationship that is required for the software to work. It is considered complex but provides the flexibility, power for the future and allow to

defined concepts in many ways.

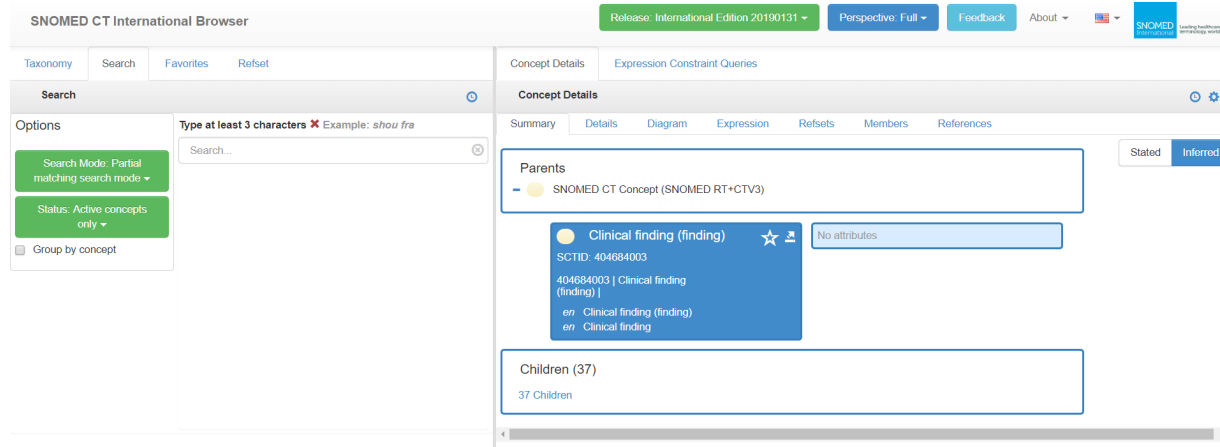


Figure 2.20: SNOMED Browser

2.6.4 Structure of SNOMED

The structure of SNOMED consists of four primary components.

1. Concept Codes: It defines the numerical codes, clinical terms, primitive or defined.
2. Description: represents the text description of the concept codes.
3. Relationship: shows the relationship between codes.
4. Reference Sets: defines the group of concepts or description, include a reference to other classification standards.

I have explained the basic structure and importance of clinical terminology. As we are using SNOMED codes with FHIR I will discuss features of SNOMED .

1. The consistency in sharing the information across healthcare systems.
2. Using SNOMED data can be organized, analysed and queried for the benefits of research and decision making.
3. Remove language barrier support multiple languages.
4. Large community to give support and updated twice a year to help users with the advancements of health care terminology.

Below figure illustrates the overall design of SNOMED

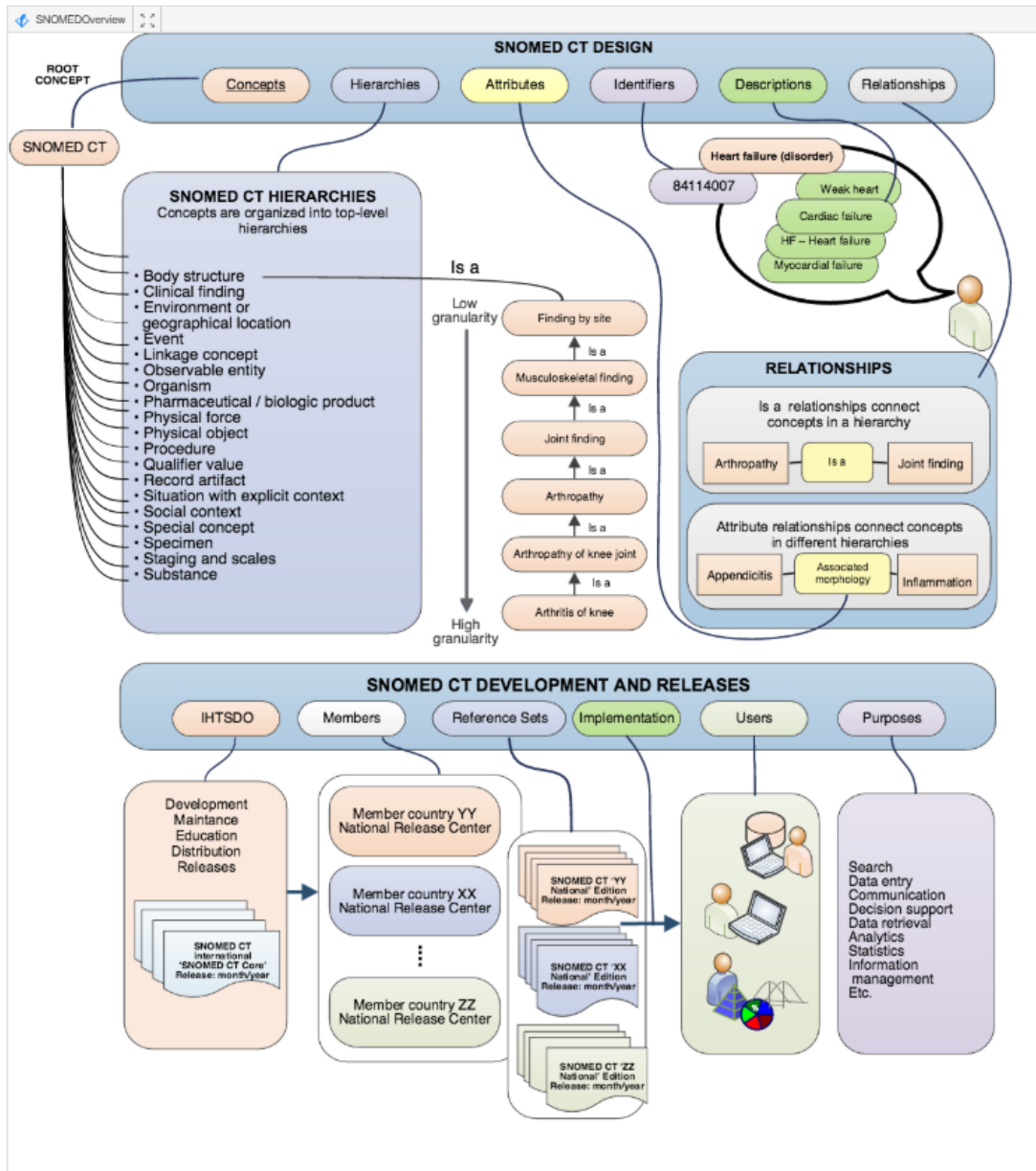


Figure 2.21: SNOMED Design

2.7 FHIR with SNOMED

In this module, we discuss requirements to use SNOMED and role that it can play in FHIR. In FHIR resources many elements have a Coded Value: with some strings or sequence of characters that identified the Concept and sequence of character can be defined in several places. By combining all these defining code they represent the Code System. It is quite sophisticated to represent the code and there are many different ways to define them [33].

According to the Specification of FHIR these coded values are composed of code and system where the system defines the URL which identifies the code system (like SNOMED). The system values are case sensitive.

To represent the coded elements in the FHIR framework following four elements are used [33].

- System: A URI that identifies the system
- Version: defined the version of the system
- Code: Pattern of strings or sequence number that identifies a concept of the code system
- Display: description of the concept defined by the code system

```
],
  "text": "low"
},
"bodySite": {
  "coding": [
    {
      "system": "http://snomed.info/sct"
      "code": "368209003",
      "display": "Right arm"
    }
  ]
}
```

Figure 2.22: SNOMED Code Example

Note that about version: In SNOMED there is no specific single distribution that defines all the codes in all context of use. As an alternative International Edition contains all the

concept that are agreed internationally. Following syntax describes how to reference the particular version of SNOMED [33].

`http://snomed.info/sct/[sctid]/version/[YYYYMMDD]` [sctid] is the concept that represent the SNOMED edition and YYYYMMDD is the date of release. If the date is not mention in the code, then terminology service affiliate the code to the most recent version of SNOMED.

The table below lists a number of national and international SNOMED CT editions.

SNOMED CT Edition	Module SCTID	Edition URI
International edition	900000000000207008	http://snomed.info/sct/900000000000207008
Australian edition (with drug extension)	32506021000036107	http://snomed.info/sct/32506021000036107
Belgian edition	11000172109	http://snomed.info/sct/11000172109
Canadian English edition	20621000087109	http://snomed.info/sct/20621000087109

Figure 2.23: SNOMED Versions [35]

2.8 Architecture of HAPI

Hapi-FHIR is developed by the University Health Network (UHN) which is java implementation for FHIR specifications. Hapi is a simple and powerful library to add FHIR messaging capabilities to our application. The main intent to design the Hapi is to provide a flexible way of using FHIR [36]. The figure below illustrates the architecture that Hapi-Fhir supports and fit to our implementation.

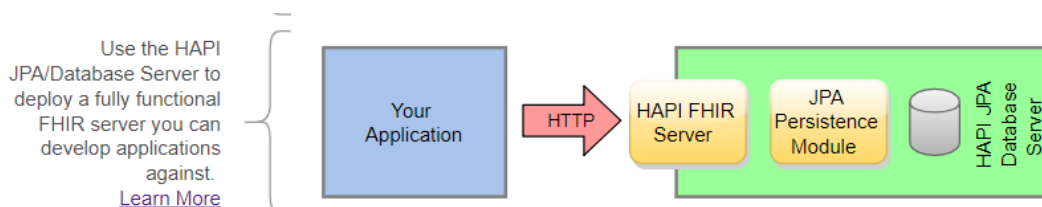


Figure 2.24: Hapi Architecture [36]

- HAPI FHIR use encoder and parser to convert between your application data model and FHIR

- HAPI FHIR client is used to store and fetch resources to an external server.
- HAPI FHIR server can be used as an application that allows an external application to modify your application.
- HAPI JPA database server can be used to deploy fully functional FHIR server and develop your application against it.

2.9 Technologies

2.9.1 JSON

In general, JSON is a lightweight data exchange format which is easy for the human to read and write. Furthermore, it is easy for the machine to parse and generate code. JSON is built on two structures. A collection of value/pair which can be an object and ordered list of values which can be an array. This is a universal data structure and all programming languages support them.

An object is an unordered set of value pair, it begins with a left brace and ends with a right brace. Every name followed the (colon) : convention and separated by the comma e.g., "name: "value. While the array is an ordered set of values, it begins with left bracket and ends with right bracket. Values are separated by a comma e.g., ["value, "value2].

As we know Restful APIs support JSON that means FHIR API is not only easy to implement in server to server communication but also underpin client-side browser applications.

2.9.2 Node.js

Node.js is an essential part of the development of our project. It enables the runtime environment for the JavaScript code. We downloaded the node.js from the official

website. After the installation is complete, we can check the version of installed node.js using the command line terminal by running the command (run node -v).

2.9.3 Web Development

For the development, we used the VUE.JS which is an open-source JavaScript framework used to build the single page application also known as the progressive framework. The web development defines the environment which has server and client, defines the server-side interaction. All the files related to the business logic or user interface are stored on the web server side. On a web browser, the user is able to render the web content by sending the request to the web server. For the communication between the HAPI server and our application, we used the JavaScript external library known as Axios, which is used to perform HTTP request that works in the browser.

2.9.4 Apache Tomcat Server

The Apache Tomcat software is an open source implementation of the Java Servlet, Pages, Java Expression Language, JavaServer and Java WebSocket technologies [40]. Implementation of a solution, we used the Tomcat server as follows. The first case is to deploy the HAPI FHIR server to receive and search the clinical data. For the second we are going to deploy another server for our web application named Development Server. It would be used to send and query the clinical data (SNOMED) from the client side. In this thesis, all the software components are build by using the JavaScript programming language.

Chapter 3

System Design

In this section, we introduce the system design for the communication of our web-application with HAPI-FHIR server. The figure below illustrates our design, on the right side our client application is shown which send the FHIR messages, retrieve data and query the HAPI FHIR server.

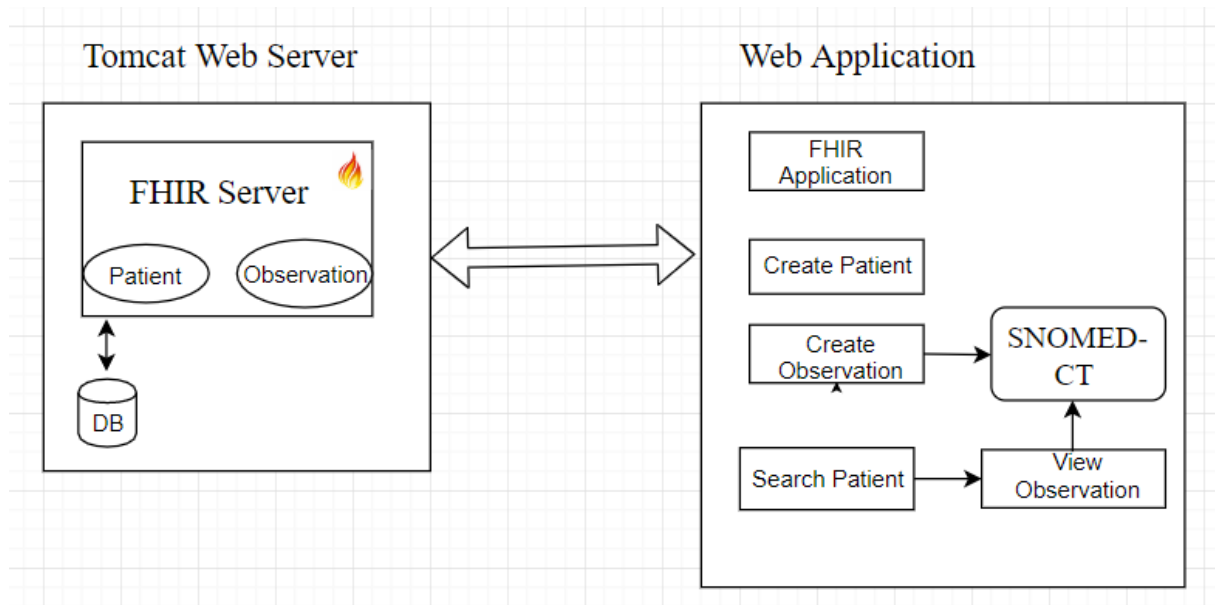


Figure 3.1: System Architecture

3.1 Client Web Application Architecture

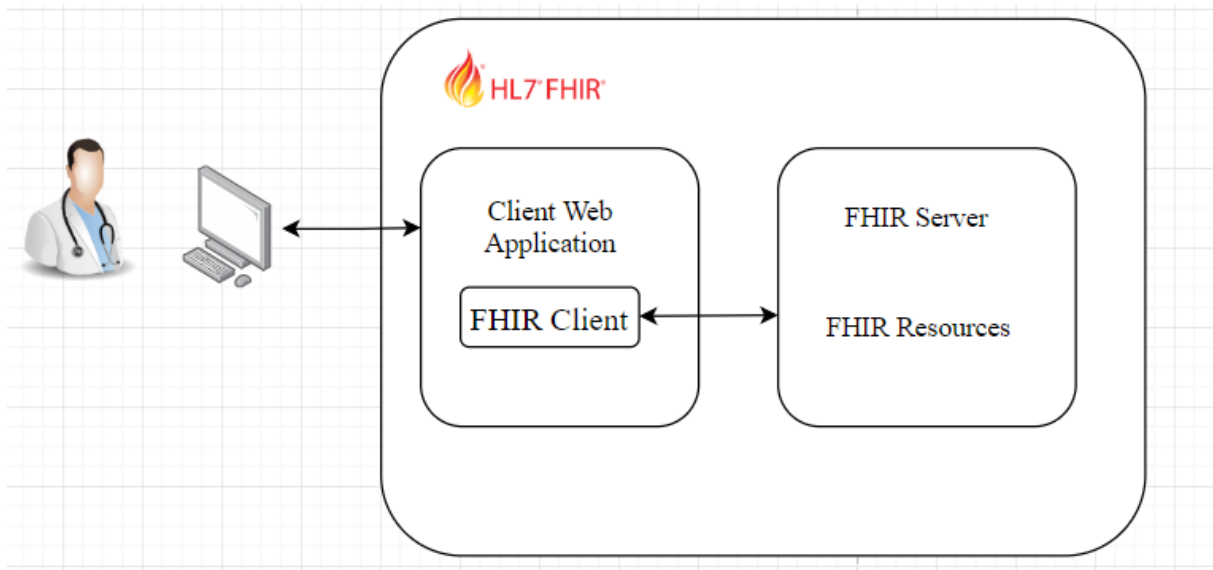


Figure 3.2: Client Web Application Architecture

3.2 Used Tools

Following is the list of tools we have used to build our application.

1. Tomcat Apache Server 9.0.0 (to host HAPI FHIR server)
2. JavaScript programming language
3. FHIR standard (First Normative Content)
4. VeuJS (Framwork)
5. Dev Server (To host our Application)
6. Windows 10 operating system
7. HTML/CSS (design our app)
8. Visual Code (IDE)
9. Apache Maven

3.3 Defining HL7 FHIR Structure for profile

In this section, we explain the FHIR profile for the Patient and Observation and other resources and make the relation between them. We used the CliniFHIR tool to validate our profile. By performing this validation steps, we make sure that we are not violating any FHIR specification. If some mistake happens in making the profile, our HAPI FHIR server will not able to communicate with our application. The figure below illustrates the elements we included in our FHIR messages.

ELEMENT	Data Type	Description
Identifier	Identifier	A unique identifier assigned to this observation.
basedOn	Reference Care	A plan, proposal or order that is fulfilled in whole or in part by this event.
status	code	The status of the result value.
category	codeableConcept	A code that classifies the general type of observation being made.
code	CodeableConcept	Describes what was observed. Sometimes this is called the observation "name".
Subject	Reference Patient	The patient, or group of patients, location, or device whose characteristics (direct or indirect) are described by the observation and into whose record the observation is placed. Comments: Indirect characteristics may be those of a specimen, fetus, donor, other observer (for example a relative or EMT), or any observation made about the subject.
Effective[x]	dateTime	The time or time-period the observed value is asserted as being true. For biological subjects - e.g. human patients - this is usually called the "physiologically relevant time". This is usually either the time of the procedure or of specimen collection, but very often the source of the date/time is not known, only the date/time itself.
bodySite	CodeableConcept	Indicates the site on the subject's body where the observation was made (i.e. the target site). Here we can include the SNOMED code
Component	Backbone Element	Some observations have multiple component observations. These component observations are expressed as separate code value pairs that share the same attributes. Examples include systolic and diastolic component observations for blood pressure measurement and multiple component observations for genetics observations.
Code	CodeableConcept	Describes what was observed. Sometimes this is called the observation "code". Add the SNOMED code here
Value	Quantity	The information determined as a result of <u>making the observation</u> , if the information has a simple value.

Figure 3.3: CliniFHIR Structure of FHIR message

By following the specification of FHIR using CliniFHIR we created the JSON data for the observation resource which contains the SNOMED code. The figure below illustrates the basic structure of the FHIR JSON template.

```

"resourceType": "Observation",
  "id": "blood-pressure",
  "meta": {
    "profile": [
      "http://uieHealth/fhir/StructureDefinition/vitalsigns"
    ]
  },
  "status": "draft",
  "category": [
    {
      "coding": [
        {
          "system": "http://terminology.hl7.org/CodeSystem/observation-category",
          "code": "vital-signs",
          "display": "Vital Signs"
        }
      ]
    }
  ],
  "code": {
    "coding": [
      {
        "system": "http://snomed.info/sct",
        "code": "75367002",
        "display": "Blood pressure panel with all children optional"
      }
    ],
    "text": "Blood pressure systolic & diastolic"
  },
  "subject": {
    "reference": "Patient/example"
  },
  "effectiveDateTime": "2018-07-02",
  "bodySite": {
    "coding": [
      {
        "system": "http://snomed.info/sct",
        "code": "368200903",
        "display": "Right Arm"
      }
    ]
  },
  "component": [
    {
      "code": {
        "coding": [
          {
            "system": "http://snomed.info/sct",
            "code": "271649006",
            "display": "Systolic Blood Pressure"
          }
        ]
      },
      "valueQuantity": {
        "value": "170",
        "unit": "mmHg",
        "system": "http://unitsofmeasure.org",

```

Figure 3.4: Observation Created using Clinifhir

3.4 Resource Handling

In resource handling we describe the way to create, update and search and history of the related resources. All the implementation is done according to the FHIR API specification. The figure below illustrates the design to create entries in the HAPI-FHIR server using our web application

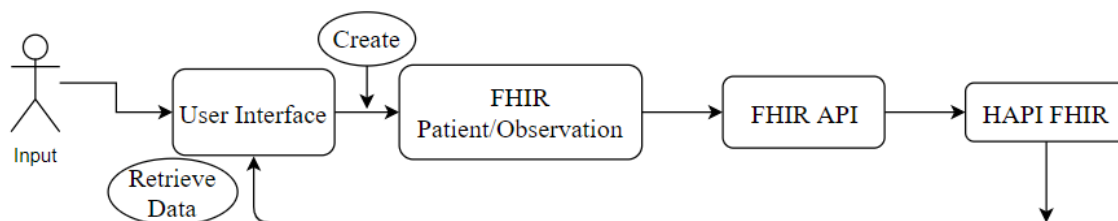


Figure 3.5: Design to create and search Resources

For the user, a web interface is introduced which provide the human readability and ability to search for specific patient observation including SNOMED codes. Creating entries for each resource is done in many steps. First, the patient data is required to enter in the resources fields which we created according to FHIR specifications. After that FHIR API sent the data to the HAPI FHIR server using the POST method. Similarly, to read and search the resources we used the GET and query method. In the process, we make the validation for the specific resource, the act of validation is like a creating. The major difference is that the user makes a validation for the particular resource using the post method to the server and receive the response from the server.

3.4.1 Query for SNOMED-CT

There are two cases first is to get data of specific patient data from the required resource. For example, if we want to read the data from our web application following syntax would be used “**GET**[baseUrl]/Patient/id”. As a response, it would return Patient resource for a specific patient with the logic [id]. Note all the request should contain a valid Authorization header and Accept Header. The Accept Header shows the format of the response

that the client is able to understand, it could be the following (application/json+fhir). A successful request received the response with code 200(successful operation) and if the request is not successful response would be (404 resource not found).

In another case we queried the HAPI FHIR server to get SNOMED-CT codes from the observation resource. Following syntax would be used

“**Get[baseUrl]/Observation?code=snomed.info/sct**”, as a result we will get all SNOMED-CT codes. To search for the specific SNOMED-CT code following syntax is used

“**Get[baseUrl]/Observation?code=snomed.info/sct/typeSNOMEDcode**”

Chapter 4

Solution Deployment pre-requisite

The goal behind this thesis is to build the application based on the HL7 FHIR model and add the SNOMED terminologies in FHIR messaging. In this chapter we describe in detail about phases we have gone through, the solution we have developed and the tools we have used in order to accomplish our goal SNOMED on FHIR. The implementation process describe in the step by step following the sequence of operation and visual illustration can be provided with screenshots along with the comment of the code snippet.

In this thesis, we have adopted the model view approach also known as (MVVM) to construct our web-application. It connects the Model and View via two-way bindings by using the JavaScript programming language and provide us the mechanism to communicate with the HAPI FHIR server. The word Model is the main component which defines the logic, data, rules and it is independent of user interfaces. The View word describes what user can see usually contains the text and other elements.

In the implementation, a use-case is adopted in which we have created the data for vital sign represented by SNOMED terminology and exchange the data between our client and HAPI FHIR server according to HL7 FHIR specification. Furthermore, we have created the functionality in our web-application to create and search Observation and Patient data.

4.1 HAPI FHIR Server

Hapi is the HL7 application programming interface (API) for Java pronounced Happy. Hapi is build to provide the design flexibility, way of adding the FHIR capabilities and Restful services. The project is feature rich including, community support, the server, excellent documentation and adding support to the newer version of the FHIR standards. The Hapi Fhir server library supports Java version 6 and easily integrated with the newer version of the JAVA. Several open-source projects build using Hapi Fhir server for example FHIR BROKER developed by National Institute of Health (NIH) and Radiology Society of North America (RSNA) image share Network, which uses the Restful calls and breaks them into existing PACS using traditional DICOM [37].

The Hapi Fhir project is developed by the University Health Network (UHN), which is a large teaching hospital in Canada. The project is open-source with a very generous Apache license which gives the user the freedom to use and distribute the modified version of the software.

In this section, we describe the procedure to setup the Hapi Fhir server which is required to host data and retain them for a specific period. It Is worth mentioning UHN provides a free public test server which is useful for a quick reference [38].

4.2 Building HAPI FHIR Environment

4.2.1 Prerequisites

Java Development Kit

To make the development environment ordinary Java Runtime Edition (JRE) is not enough. We are required to install the Java Development Kit (JDK) to compile the java code which is necessary to build the FHIR server. There are two possibilities for JDK, one provided by the Oracle <http://www.oracle.com/technetwork/java/javase/downloads/index.html>)

and other option is OpenJDK (<http://openjdk.net>). We are using the Windows operating system, the installation of JDK is done via an application store from the official website of JDK provider. We have installed the latest version of JDK and it is recommended to installed the newest version to avoid any problem.

Java Application Server

For the development of our application, Java application server is required. Different servers are available like Apache Tomcat, Websphere, JBoss. For our thesis, we are focused on Tomcat version 9. We have installed the Tomcat server using Windows Installer Service from official website <http://tomcat.apache.org/download>, which install the server on the system service that makes it easier to stop or start the server using Control Panel Admin tools. After completing the installation we can see the running server with an index page at <http://localhost:8080> seen in the figure below.

The screenshot shows the Apache Tomcat 9.0.14 index page. At the top, there is a navigation menu with links for Home, Documentation, Configuration, Examples, Wiki, and Mailing Lists, along with a Find Help button. The main heading is "Apache Tomcat/9.0.14" with the Apache Software Foundation logo. A green banner reads "If you're seeing this, you've successfully installed Tomcat. Congratulations!". Below this, there is a "Recommended Reading" section with links to "Security Considerations HOW-TO", "Manager Application HOW-TO", and "Clustering/Session Replication HOW-TO". To the right of these links are buttons for "Server Status", "Manager App", and "Host Manager". A "Developer Quick Start" section contains links for "Tomcat Setup", "Realms & AAA", "Examples", and "Servlet Specifications". The page is divided into three main content boxes: "Managing Tomcat" (with links for Release Notes, Changelog, Migration Guide, Security Notices), "Documentation" (with links for Tomcat 9.0 Documentation, Tomcat 9.0 Configuration, Tomcat Wiki, and various developer resources), and "Getting Help" (with links for FAQ and Mailing Lists and a list of available mailing lists).

Figure 4.1: Index Page of Tomcat server

Apache Maven (Java Package Manager)

Apache Maven is the software management tool for Java-based projects. The primary use of maven is to build the project, dependencies and documentation. It is required to resolve the HAPI dependencies and compile entire package which can be deployed on Tomcat server. We have download the newest release of maven from (<http://maven.apache.org/download.cgi>). We have unzipped the downloaded file and put the bin directory to the operating system PATH environment variables. In our case, we have kept the default directory C:\files after that add the Path environment variables as shown in the figure.

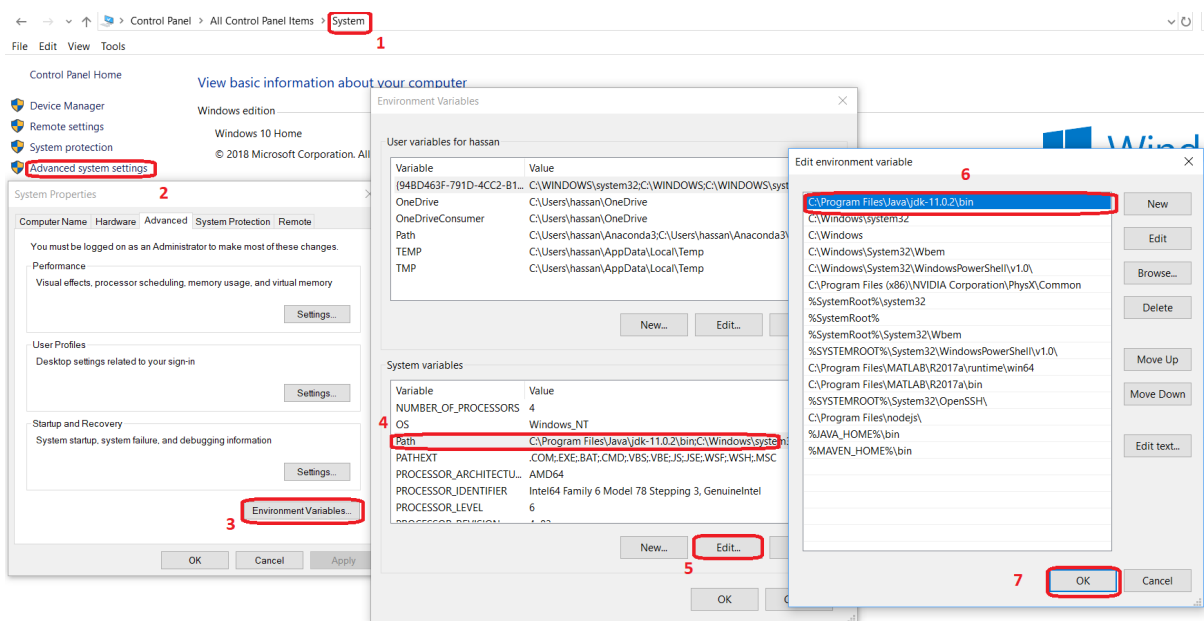


Figure 4.2: Adding Path Environment Variables

4.3 Installing HAPI FHIR

We have completed all the prerequisites to install the HAPI server. Next step is to download the newest release HAPI project from <https://github.com/jamesagnew/hapi-fhir/releases>. The file name is called hapi-fhir-jpaserver-example.zip so called JPA server. After unzipping the file from the archive open the GitBash terminal moved to the directory where the source code exists. Then run the command “mvn install” to compile the code. This process will take some time to download all the dependencies and packages. After finishing a file with an extension (.WAR) is created, now copy the (.WAR) file in the tomcat server directory. For example, C:

Program Files/Apache Software Foundation9.0. To check the HAPI FHIR server is correctly installed go to the browser and write the following URL <http://localhost:8080/hapi-fhir-jpaserver-example/>. It will take some time to load the server.

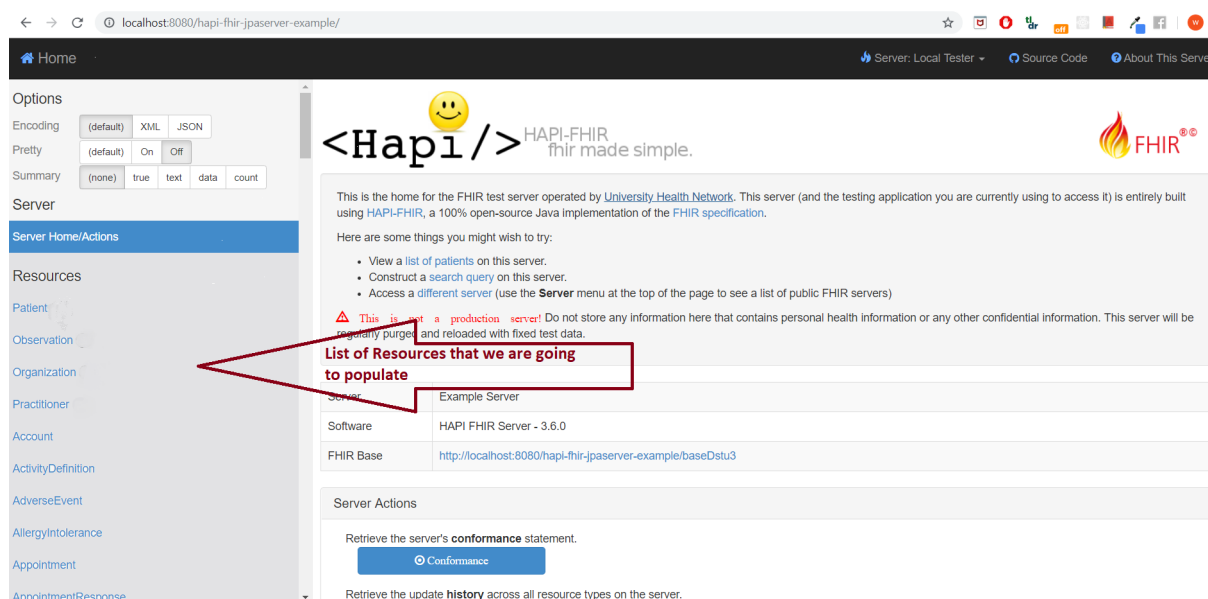


Figure 4.3: HAPI FHIR Server

4.4 Testing Hapi API with Postman

After installing the HAPI server successfully we need to test the FHIR server API in order proceed with our development. To test the API, we have used a tool called Postman. Postman is a powerful tool used to interact with HTTP API. To send the data we have to use the base URL of the HAPI server named as `http://localhost:8080/hapi-fhir-jpaserver-example/baseDstu3`. In order to send the data to a specific resource, we need to give the resource path by adding the name of a resource at the end of the base URL of the HAPI server API. We have created the data as Patient observation and we have successfully Post/Get the data to the HAPI FHIR API and it supports all the CRUD (Create, Read, Update, Delete) operations. The figure below illustrates the success of sending the Observation data to the resource name Observation.

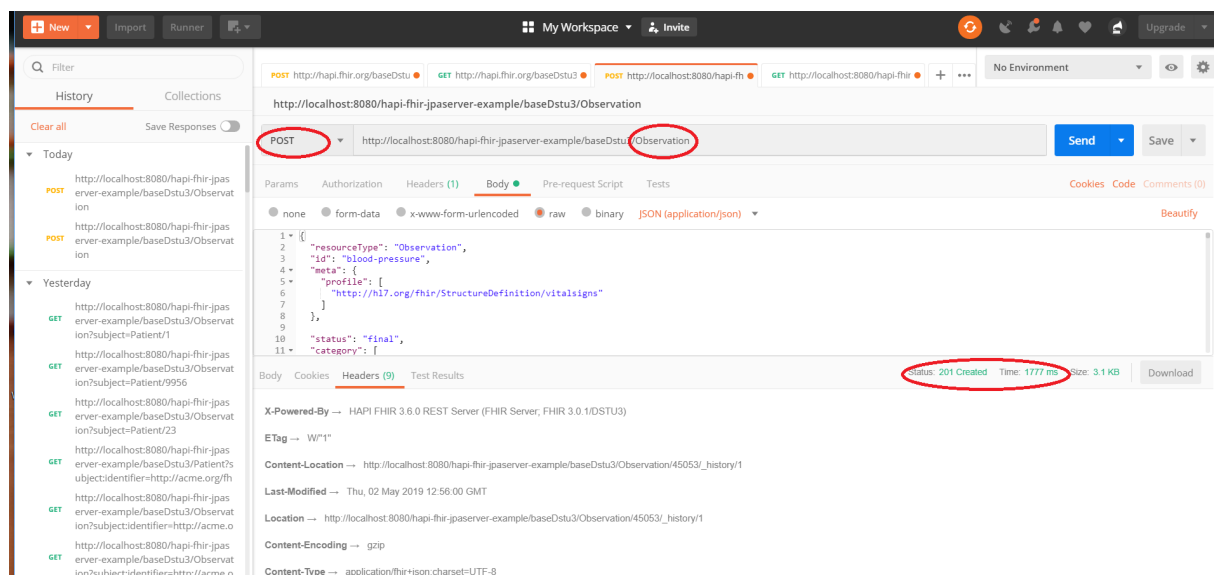


Figure 4.4: Postman sending Post request to FHIR server

Chapter 5

Solution Implementation

In this section, we implement the required functionalities for the Front-end and the back-end of our web application. The web application can perform the following operations against HAPI FHIR server.

1. Create a Patient and Observation statement and send to HAPI FHIR server.
2. Read Patient observation statement from HAPI FHIR server.
3. Search for SNOMED code for a specific patient.
4. Search all the SNOMED code.

In our web application, we have developed the process for creating the resources, enabled the communication with the HAPI FHIR server and provided the user interface for the client to search for the SNOMED code with patient details. All the requirements needed to setup the environment have discussed in the previous chapter.

5.1 Project Strucutre

As we know Veu.js framework used the component-based approach in order to structure the project. Vue.js uses the HTML bases template which allow us to bind the rendered

DOM (Document Object Model) with underlying Vue instance data. Vue template uses the Virtual DOM to render the defined functions. It is intelligent enough to figure out the minimal number of components need to render and apply the DOM manipulation when state of the application changes.

We have structure our project according to their roles the structure overview can be seen in the figure below

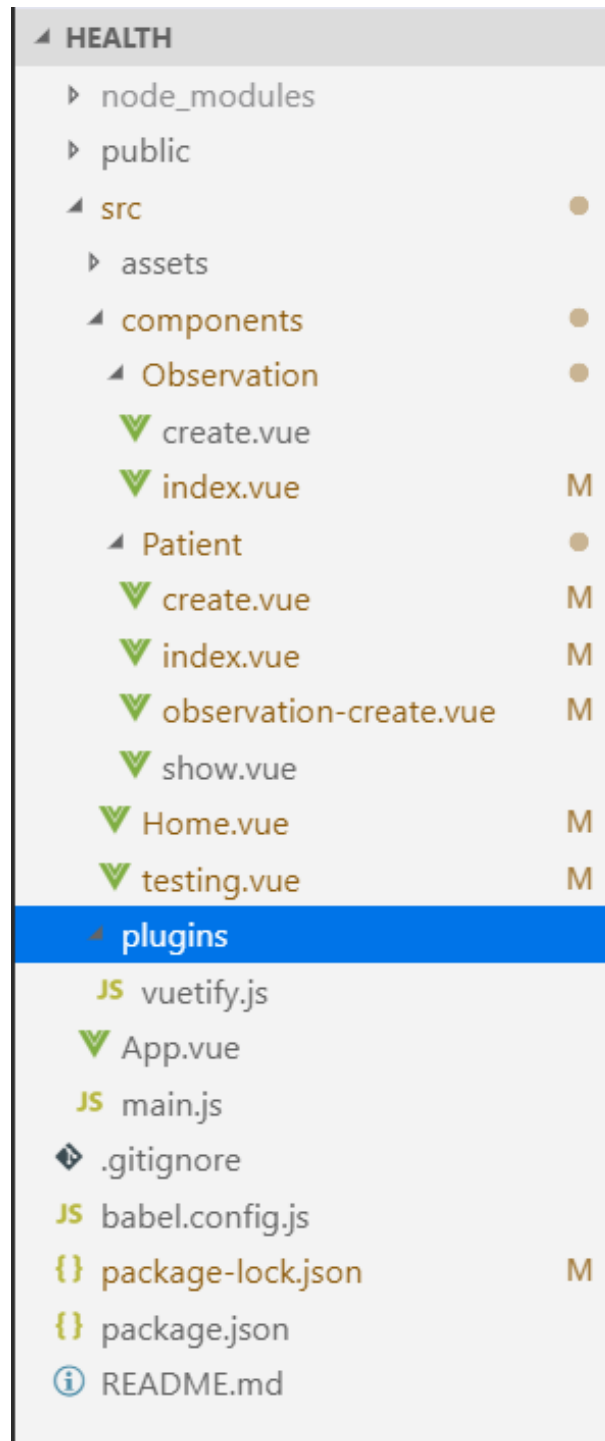
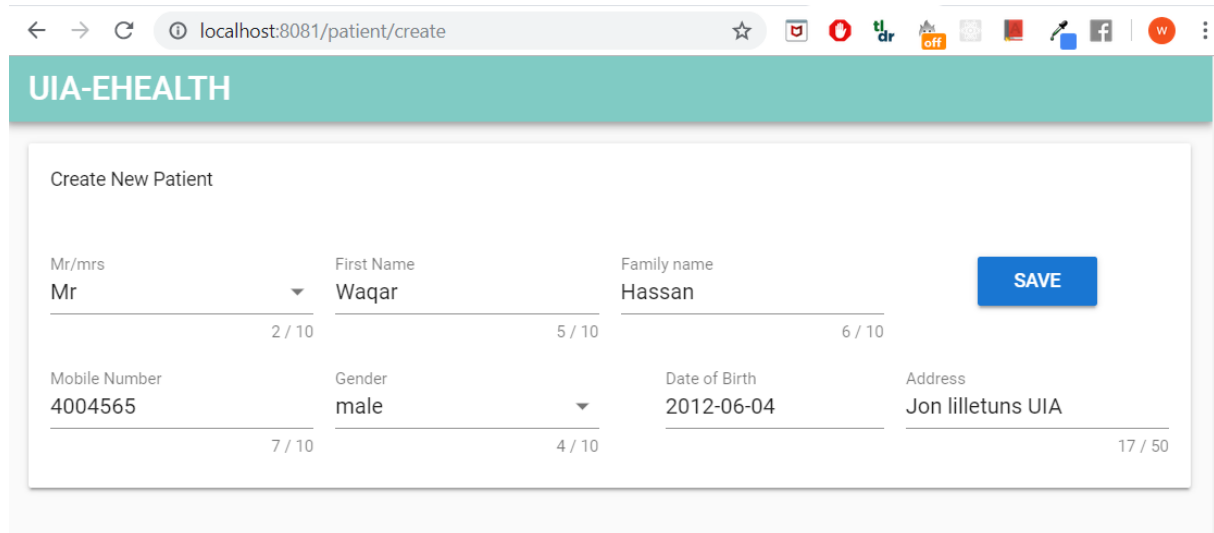


Figure 5.1: Project Structure

5.2 Create Patient Record

In this section we describe the process of creating the FHIR enable patient record, using it we can send the information to HAPI FHIR server. The figure below illustrates the procedure to create a new patient.



The screenshot shows a web browser window with the URL `localhost:8081/patient/create`. The page title is "UIA-EHEALTH". The main content is a form titled "Create New Patient". The form has the following fields and values:

Field	Value	Character Count
Mr/mrs	Mr	2 / 10
First Name	Waqar	5 / 10
Family name	Hassan	6 / 10
Mobile Number	4004565	7 / 10
Gender	male	4 / 10
Date of Birth	2012-06-04	
Address	Jon lilletuns UIA	17 / 50

A blue "SAVE" button is located to the right of the form.

Figure 5.2: Create Patient Record

When the client clicks on the save button DOM event is created and called the `SavePatient` method we have created shown in the snippet below.

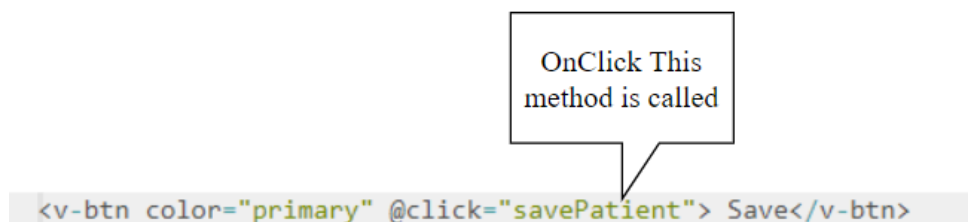


Figure 5.3: OnClick Function called Named SavePatient

In this method we defined the FHIR message which take the value dynamically from the client and send it to the HAPI server illustrated in the snippet below.

```

methods: {
  savePatient() {
    let data = {
      "resourceType": "Patient",
      "id": "1",
      "identifier": [
        {
          "use": "usual",
          "type": {
            "coding": [
              {
                "system": "http://terminology.hl7.org/CodeSystem/v2-0203",
                "code": this.mr
              }
            ]
          },
          "system": "uia.no",
          "value": "12345",
          "assigner": {
            "display": "UIA Health"
          }
        }
      ],
      "active": true,
      "name": [
        {
          "use": "official",
          "family": this.familyname,
          "given": [
            this.firstname
          ]
        }
      ],
      "telecom": [
        {
          "use": "home"
        }
      ]
    }
  }
}

```

Method Created

FHIR Message

Figure 5.4: SavePatient Method Contains the FHIR messages

We have successfully created the FHIR patient now our next step is to send the data to the HAPI server following snippet is used to send the patient details.

```

this.axios.post(this.baseUrl + 'Patient', data).then((response) => {
  console.log(response.data);
  this.$notify.success("Successfully saved!");
  this.$router.push({name: 'patients'});
})

```

JavaScript Library/
XMLHttpRequests

HTTP Method

URL of HAPI FHIR server

Route to the HAPI server Resource

Contain the FHIR data

Server Response

The data is successfully created, and we have received the response with status code 200. The figure below illustrates the server response and the patient resource has received the data.



← → ↻ localhost:8080/hapi-fhir-jpaserver-example/baseDstu3/Patient/65056/_history/1

This result is being rendered in HTML for easy viewing. You may access this content as [Raw JSON](#) or [Raw XML](#), or in 6ms.

HTTP 200 OK

Response Headers

X-Powered-By: HAPI FHIR 3.6.0 REST Server (FHIR Server; FHIR 3.0.1/DSTU3)

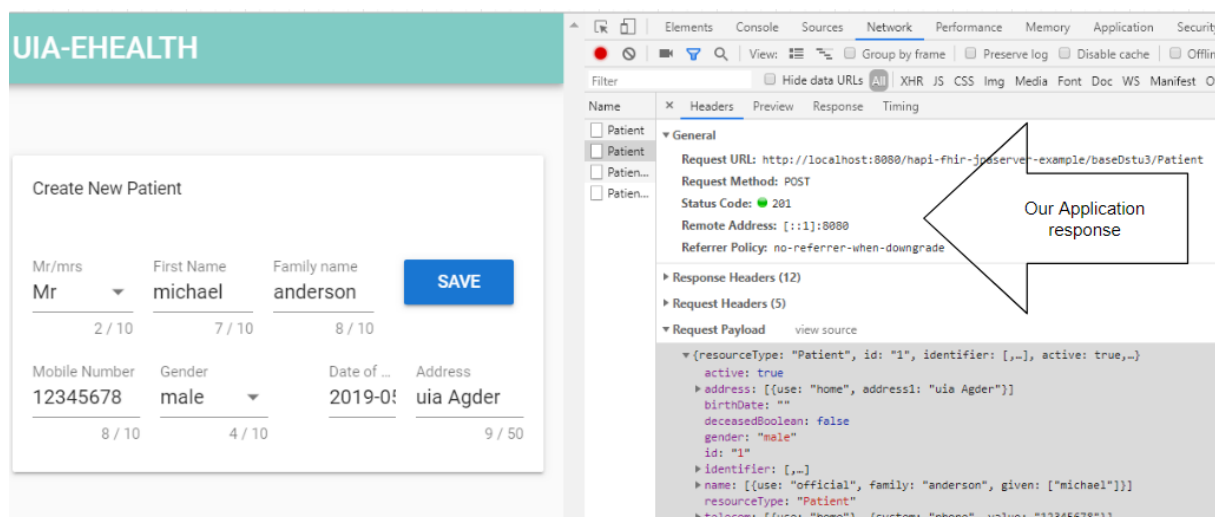
Response Body

```

1  {
2    "resourceType": "Patient",
3    "id": "65056",
4    "meta": {
5      "versionId": "1",
6      "lastUpdated": "2019-05-23T04:17:50.267+02:00"
7    },
8    "text": {
9      "status": "generated",
10     "div": "<div xmlns=\\"http://www.w3.org/1999/xhtml\\"><div class=\\"hapiHeaderText\\">michael <b>ANDERSON
11     },
12     "identifier": [
13       {
14         "use": "usual",
15         "type": {
16           "coding": [
17             {
18               "system": "http://terminology.hl7.org/CodeSystem/v2-0203",
19               "code": "Mr"
20             }
21           ]
22         },
23         "system": "uia.no",
24         "value": "12345",
25         "assigner": {
26           "display": "UIA Health"

```

Figure 5.5: HAPI FHIR server response



UIA-EHEALTH

Create New Patient

Mr/mrs: Mr, First Name: michael, Family name: anderson, SAVE

Mobile Number: 12345678, Gender: male, Date of Birth: 2019-01, Address: uia Agder

Network Tab: Headers, Response, Timing

Request URL: http://localhost:8080/hapi-fhir-jpaserver-example/baseDstu3/Patient

Request Method: POST

Status Code: 201

Remote Address: [::1]:8080

Request Payload:

```

{resourceType: "Patient", id: "1", identifier: [,-], active: true,-}
  active: true
  address: [{use: "home", address1: "uia Agder"}]
  birthDate: ""
  deceasedBoolean: false
  gender: "male"
  id: "1"
  identifier: [,-]
  name: [{use: "official", family: "anderson", given: ["michael"]}]]
  resourceType: "Patient"
  telecom: [{use: "home"}, {system: "phone", value: "12345678"}]

```

Figure 5.6: Our Application Response

Figure below illustrates the FHIR resources are populated by our data.

The screenshot displays a web interface for a FHIR server. On the left, a sidebar lists various resource types: Patient (66), Observation (50), Organization (1), Practitioner (1), Account, ActivityDefinition, and AdverseEvent. A large arrow points from the text 'Resources populated by our data' to the Observation resource. On the right, there is a warning message: 'This is not a production s regularly purged and reloaded with t'. Below this is a table with the following data:

Server	Example Serve
Software	HAPI FHIR Ser
FHIR Base	http://localhost:

Below the table, there is a section titled 'Server Actions' with two buttons: 'Conformance' and 'History'.

Figure 5.7: FHIR resource populated

5.3 Create SNOMED-CT for Blood Pressure

In this section, we have created the Observation form to record the vital-signs (blood-pressure) including the SNOMED code. We used SNOMED browser to get the codes for a vital sign we are measuring. We can mention the specific patient while creating a new Observation. The figure below illustrates the form, created to send the clinical data to the HAPI FHIR server. As a result, new instance is created observation resource which contains the SNOMED-CT terminologies of the blood pressure. In observation form, we have made option to search the patient in order to give reference to the specific patient

UIA-EHEALTH

Create New Vital Sign Observation for Blood Pressure

Select the Patient Select Date

Patient Name Date

SNOMED code for BodySite SNOMED code for BodySite SNOMED description

9 / 15 9 / 15

SNOMED Code Systolic BP Systolic BP Systolic Value of Blood Pressure

9 / 30 0 / 15

Code Diastolic SNOMED Description Diastolic Value of Blood Pressure

9 / 30 0 / 15

SAVE

Figure 5.8: Observation Form

The communication between our application and HAPI FHIR server is happened in the same way we discussed earlier, but in this case the route is changed to Observation path as shown in the snippet below.

```

this.axios.post(this.baseUrl + 'Observation', data).then((response) => {
  console.log(response.data);
  this.$notify.success('Successfully added');
  this.$router.push({name: 'patient-show'});
});

```

```

baseUrl: 'http://localhost:8080/hapi-fhir-jpaserver-example/baseDstu3/',

```

Figure below shows the SNOMED-CT terminologies are created for blood pressure in HAPI FHIR server.

```

1  {
2    "resourceType": "Observation",
3    "id": "50054",
4    "meta": {
5      "versionId": "1",
6      "lastUpdated": "2019-05-03T00:57:35.429+02:00",
7      "profile": [
8        "http://hl7.org/fhir/StructureDefinition/vitalsigns"
9      ]
10   },
11   "status": "final",
12   "category": [
13     {
14       "coding": [
15         {
16           "system": "http://terminology.hl7.org/CodeSystem/observation-category",
17           "code": "vital-signs",
18           "display": "Vital Signs"
19         }
20       ]
21     }
22   ],
23   "code": {
24     "coding": [
25       {
26         "system": "http://snomed.info/sct",
27         "code": "75367002",
28         "display": "Blood pressure panel with all children optional"
29       }
30     ]
31   },
32   "text": "Blood pressure systolic & diastolic"
33 },
34 "subject": {
35   "reference": "Patient/50052"
36 },
37 "effectiveDateTime": "2019-05-04",
38 "bodySite": {
39   "coding": [
40     {
41       "system": "http://snomed.info/sct",
42       "code": "368209003",
43       "display": "Right arm"
44     }
45   ]
46 },
47 "component": [
48   {
49     "code": {
50       "coding": [
51         {
52           "system": "http://snomed.info/sct",
53           "code": "271649006",
54           "display": "Systolic blood pressure"
55         }
56       ]
57     },
58     "valueQuantity": {
59       "value": 89,
60       "unit": "mmHg",
61       "system": "http://unitsofmeasure.org",
62       "code": "mm[Hg]"
63     }
64   }
65 ]
66 }

```

Figure 5.9: SNOMED-CT created in HAPI FHIR resource

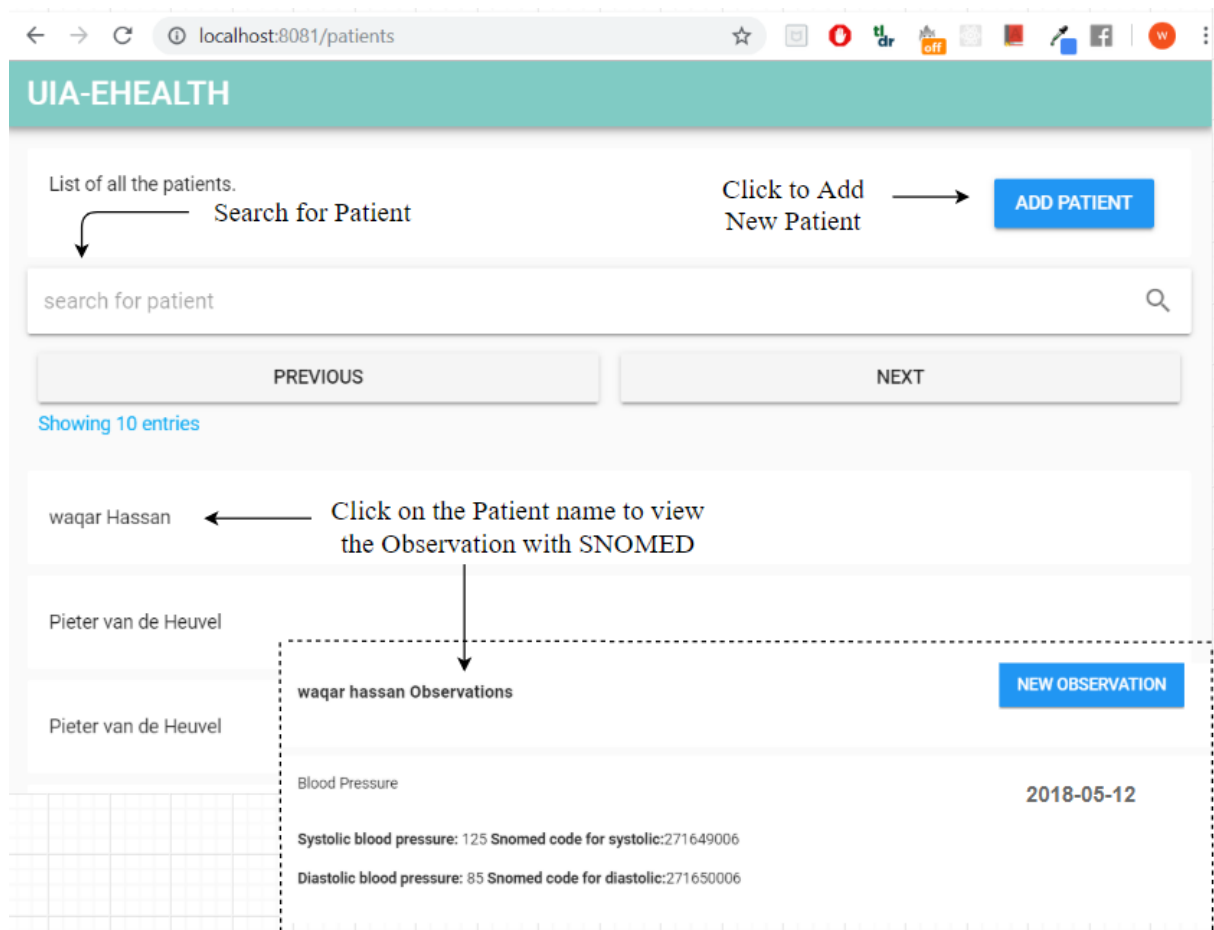
5.4 Read SNOMED-CT

To read the all the SNOMED-CT from the HAPI FHIR server we have created a function in our application. In which we have used HTTP “Get” method to retrieve the SNOMED terminologies. The following snippet code is used to read and retrieve information for the HAPI FHIR server.

```
methods:{
  // Using this snippet we will get all the SNOMED-CT codes stored in the HAPI FHIR server
  // baseUrl: http://localhost:8080/hapi-fhir-jpaserver-example/baseDstu3/
  //Path: Observation
  // ? : used as a search operator
  getPatient(){
    this.axios.get(this.baseUrl+'Observation?code=snomed.info/sct').then((response) => {
      console.log(response.data);
      this.observations = response.data.entry
    })
  }
}
```

5.5 SNOMED Search

By default, HAPI server provides is the parameter for the search and filter the resource instance. So, we can do a search by code, language and HL7 FHIR standards are used to search for the specific data (for example id, query, tag, filter etc.). These are the parameter which we can use query the HAPI FHIR server. The search framework defined by the HL7 provides the search based on index criteria. The word query gives us the control to do more precise, complex decision support-based queries that have human resolution [41]. In our application, we have design interface for the user where a client can search the patient by his name and see the specific observation of the patient which include the SNOMED-CT code with all the description related to the symptoms. The figure below shows the UI to search for the Patient and SNOMED codes related to the patient.



The search panel is getting the data from the HAPI FHIR server. When we write the name of it queries the bundle of data, we are getting from the HAPI FHIR server and filter the data. The following snippet can be used to query.

```

data: () => ({
  baseUrl: 'http://localhost:8080/hapi-fhir-jpaserver-example/baseDstu3/',
  patients: [], // Array is created to save the list of patient data from the Hapi FHIR server
  searchText: '', // A state is created to enable the search in our application.
  offset: 0
}),
methods: {
  onSearch(){
    this axios.get(this.baseUrl + 'Patient?name=' + this.searchText).then((response) => {
      console.log(response.data);
      this.patients = response.data.entry
    });
  }
}

```

```

getPatientObservations() {
  this.axios.get(this.baseUrl + 'Observation?subject:Patient=' + this.$route.query.patient_id).then((response) => {
    console.log(response.data);
    this.observations = response.data.entry
  })
},
getPatientById() {
  this.axios.get(this.baseUrl + 'Patient?id=' + this.$route.query.patient_id).then((response) => {
    console.log(response.data);
    this.patient = response.data.entry
  })
},

```

Search for SNOMED-CT in Observation resource

Search for the Patient with Id

Filters the SNOMED-CT code to display

```

<p><b>{{observation.resource.component[0].code.coding[0].display}}:</b>
  {{observation.resource.component[0].valueQuantity.value}} <b>Snomed code for
  systolic:</b>{{observation.resource.component[0].code.coding[0].code}</p>
<p><b>{{observation.resource.component[1].code.coding[0].display}}:</b>
  {{observation.resource.component[1].valueQuantity.value}} <b>Snomed code for
  diastolic:</b>{{observation.resource.component[1].code.coding[0].code}</p>

```

UIA-EHEALTH

VIEW PATIENTS

VIEW OBSERVATIONS

To view the patients and create the Patient click on the view patient button. which opens the new page with the ability to search the patients and see the vital signs of specific patient and their SNOMED-CT codes

To view all the observation stored in the Hapi Fhir server click on the view observation button

Figure 5.10: HomePage of our Application

Chapter 6

Discussion

The main objective of the thesis is to implement SNOMED-CT terminologies using the HL7 FHIR information model. The assessment on international standards made by the Norwegian Directorate of eHealth recommended the use of Fast Health Interoperability Resources (FHIR) based on the service-oriented architecture. Furthermore, the FHIR resources can be used for the national requirement as well. Therefore, we decided to proceed with the HL7 FHIR standard as the chosen framework for our web application.

We have created the FHIR message for SNOMED-CT that represent the blood pressure. We have used the JSON notation, and their primary function is to serialize and transmit the structured data using a network connection and used to transfer data between web application and the server. At this stage, we managed to create a FHIR message to populate the resources.

The most important feature of our application that it is build according to the FHIR specification which support interoperability and scalability .The FHIR profile we created to represent the vital-sign information, we added the SNOMED-CT codes in Bodysite and Coding. In which we mentioned the system of codes, description and codes for the SNOMED. We have defined the method in our application in order to search for SNOMED-CT codes and all the implementation is done according to the FHIR specification. As a result, the client can use our solution to search the number of patients with a

specific vital sign (SNOMED code) that is implicitly offered by FHIR Restful server.

In our web application, the components we developed demonstrated a client use case, where the client is able to send a request to HAPI FHIR server and obtain information about vital-signs with SNOMED-CT. For example, by using the application the client can create vital-signs FHIR messages and store them in the HAPI FHIR server and our application provide the ability to search. Consequently, the client is able to extract specific patient information.

We have defined the SNOMED-CT terminologies in the “coding” component of the FHIR message. The “coding” component contains three elements: system, code and display. “System” represent the SNOMED-CT terminology, “code” contains the actual SNOMED code and “display” represent the human readable description of the code.

We have query the SNOMED-CT codes in which HAPI FHIR server returns all the SNOMED-CT codes related to Blood pressure.

In this thesis, details in the implementation of solution reflects the efforts and challenges I have faced and overcome during the implementation phase. One of the challenges I face was the lack of the documentation available to deploy the HAPI FHIR JPA server. I had to go through all the FHIR documentation, HAPI java documentation and external documentations in order to understand and implement our solution.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

The exchange of medical information plays an essential role for patient, practitioners and researchers. The development of such a system which provide the interoperability and decision support can help to reduce the wrong medication of the patient. In health information system the standardization and structuring of information are important. We have used SNOMED-CT FHIR to provide the semantic interoperability. The use of SNOMED and FHIR potential strengths can be defined from its capabilities for interaction and integration. In other words, it helps to build the bridge between systems and organizations.

In our proposed solution we developed the web application that is built on the top of FHIR standard and the clinical terminologies to represent the vital-sign terminologies. We defined the FHIR profiles for the patients. We transfer the clinical data by making a form through which client or practitioner can enter the clinical along with SNOMED terminologies. After, this clinical information is mapped on the FHIR standard and transferred using HTTP communication standard with HAPI FHIR server. Furthermore, our application gives the ability to the client to search for a specific SNOMED-CT codes.

As an overall result, the supporting arguments of this thesis are practically validated. The

vital signs clinical data created, stored, queried and accessible by using the HL7 RESTful bases service.

7.2 Future Work

At this time our application deals with a single vital sign, the system provides the tools to make the FHIR resource instances. The FHIR has a number of resources which cover the issues related to healthcare and administration. In the future work development of the application includes addition of more vital signs and add the capability to search multiple SNOMED-CT codes at the same time. In this system, two different servers are user HAPI FHIR and our application development server. The system is restricted to the functions that two servers provide. It would be beneficial if we develop a special FHIR server in order to meet our desired requirements.

Bibliography

- [1] H. I. E. (HIE), "Standards & Interoperability," HealthIT.gov, 27 February 2014. [Online]. Available:<https://www.healthit.gov/providers-professionals/standards-interoperability>. [Accessed 20 March 2019]
- [2] About Norwegian Centre for E-health Research, 28 September 2018 [Online].
- [3] Terminfo H. Term Info Available: at: <http://www.hl7.org/Special/committees/terminfo/index.cfm>, 2017 [Accessed March 2019]
- [4] K. C. Stange. The problem of fragmentation and the need for integrative solutions. *Ann Fam Med*, 7(2):100–103, 2009 [Accessed March 2019]
- [5] The open E.H.R foundation [ONLINE] Accessed "D Kalra, T Beale, S Heard - Studies in health technology and ..., 2005 - books.google.com" [Accessed March 2019]
- [6] Principles of Health Interoperability HL7 and SNOMED (Health Informatics) Available at [<https://link.springer.com/book/10.1007/978-3-319-30370-3>] [Accessed March 2019]
- [7] Robert H. Dolin, Liora Alschuler, Sandy Boyer, Calvin Beebe, Fred M. Behlen, Paul V. Biron, Amnon Shabo (Shvo), HL7 Clinical Document Architecture, Release 2, *Journal of the American Medical Informatics Association*, Volume 13, Issue 1, January 2006, Pages 30–39 [Accessed March 2019]
- [8] FHIR® – Fast Healthcare Interoperability Resources , "Summary - FHIR," Health Level Seven® INTERNATIONAL, 2018. [Online]. Available: <http://hl7.org/fhir/summary.html> [Accessed March 2019]
- [9] D. Bender and K. Sartipi, "HL7 FHIR: An Agile and RESTful approach to healthcare information exchange," *Proceedings of the 26th IEEE International Symposium on Computer-Based Medical Systems*, Porto, 2013, pp. 326-331. doi: 10.1109/CBMS.2013.6627810 [Accessed March 2019]
- [10] Principles of Health Interoperability HL7 and SNOMED Interoperability Book [Accessed March 2019]
- [11]. Hapi FHIR Server available at [<http://hapifhir.io/>] [Accessed March 2019]
- [12] FHIR® – Fast Healthcare Interoperability Resources HL7 Available at: [https://www.hl7.org/documentcenter/public_temp_7A23B0A2-1C23-BA17-0C0E29A347FE7DA5/newsroom/HL7%20EU%20response%20FINAL.pdf] [Accessed March 2019]

- [13] Evolution of HL7 standards. Available at:
<https://www.phiresearchlab.org/index.php/2016/03/25/the-evolution-of-hl7-standards-v2-to-fhir/> Fast Healthcare Interoperability Resources
- [14] Fast Healthcare Interoperability Resources HL7 “Summary FHIR” Available at
[\[https://www.hl7.org/fhir/summary.html\]](https://www.hl7.org/fhir/summary.html)
- [15] Fast Healthcare Interoperability “Development Overview of FHIR” Resources HL7
Available at: <https://www.hl7.org/fhir/overview-dev.html>
- [16] Fast Healthcare Interoperability “Data types of FHIR” Resources HL7 Available at:
<http://hl7.org/fhir/datatypes.html>
- [17] Fast Healthcare Interoperability “Architectural overview of FHIR” Resources HL7
Available at: <https://www.hl7.org/fhir/overview-arch.html>
- [18] Fast Healthcare Interoperability “Restful API FHIR” Resources HL7 Available at:
<https://www.hl7.org/fhir/http.html>
- [19] Fast Healthcare Interoperability “Profiling in FHIR” Resources HL7 Available at:
<https://www.hl7.org/fhir/profiling.html>
- [20] Fast Healthcare Interoperability Resources “Clinical overview of FHIR” HL7 Available at:
<http://hl7.org/fhir/overview-clinical.html>
- [21] Fast Healthcare Interoperability “Extensions on FHIR” Resources HL7 Available at:
<https://fhirblog.com/2017/04/12/so-you-want-to-create-a-fhir-extension/>
- [22] Fast Healthcare Interoperability “Reference in FHIR” Resources HL7 Available at:
<https://www.hl7.org/fhir/references.html>
- [23] Fast Healthcare Interoperability “Terminology Module in FHIR” Resources HL7 Available
at: <https://www.hl7.org/fhir/terminology-module.html>
- [24] Fast Healthcare Interoperability Resources HL7 “FHIR Index page” Available at:
<http://hl7.org/fhir/index.html>
- [25] Health Care Interoperability by Graham Grieve Available at:
<http://www.healthintersections.com.au/?p=2503>
- [26] Fast Healthcare Interoperability “Vital Sign in FHIR” Resources HL7 Available at:
<http://www.hl7.org/fhir/us/core/us-core-vitalsigns.html>
- [27] Fast Healthcare Interoperability “Patient Resource” Resources HL7 Available at:
<https://www.hl7.org/fhir/patient.html>
- [28] Fast Healthcare Interoperability “Observation Resource” Resources HL7 Available at:
<https://www.hl7.org/fhir/observation.html>

[29] Fast Healthcare Interoperability “Practitioner Resource” Resources HL7 Available at: <https://www.hl7.org/fhir/practitioner.html>

[31] Clinical Terminology and clinical classification Available at: <https://perspectives.ahima.org/clinical-terminology-and-clinical-classification-systems-a-critique/>

[32] SNOMED-CT in Norway Available at: <https://ehelse.no/standarder-kodeverk-og-referanse katalog/helsefaglige-kodeverk/snomed-ct>

[33] SNOMED-CT Starter GUIDE Available at: <https://confluence.ihtsdotools.org/display/DOCSTART/4.+SNOMED+CT+Basics> [Online Accessed April 2019]

[34] Fast Healthcare Interoperability “SNOMED on FHIR” Resources HL7 Available at: <https://www.hl7.org/fhir/terminologies.html> [Online Accessed April 2019]

[35] SNOMED-CT editions Available at: <https://confluence.ihtsdotools.org/display/DOCEXTPG/4.4.2+Edition+URI+Examples> [Online Accessed April 2019]

[36] Website for HAPI FHIR server documentation Available at: <http://hapifhir.io/> [Accessed April 2019]

[37] HAPI FHIR server project on github Available at: <https://github.com/RSNA/s4s-fhir-broker> [Online Accessed April 2019]

[38] Public HAPI-FHIR server Available at [fhirtest.uhn.ca] [Online Accessed April 2019]

[39] Dobrev, A., Stroetmann, K.A., Stroetmann, V.N., Artmann, J., Jones, T., & Hammerschmidt, R. (2008). The conceptual framework of interoperable electronic health record and ePrescribing systems. [Online Accessed April 2019]

[40] Official website of Tomcat server Available at [<http://tomcat.apache.org/>] [Online Accessed April 2019]

[41] Terminfo H. Terminfo Project. Available at: <http://www.hl7.org/Special/committees/terminfo/index.cfm>, 2017 [Online Accessed April 2019]

[42] Apelon Clinical Terminology services provide Available at [<https://www.apelon.com/>] [Online Accessed April 2019]

[43] Intelligent Model Object “Enterprise Medical Terminology Platform” Available at [<https://www.e-imo.com/>] [Online Accessed April 2019]

Appendix

Application Code

```
1. import Vue from 'vue'
2. import './plugins/vuetify'
3. import App from './App.vue'
4. import Vuetify from 'vuetify'
5. import 'vuetify/dist/vuetify.min.css' // Ensure you are using css-
  loader
6. import VueRouter from 'vue-router'
7. import axios from 'axios'
8. import VueAxios from 'vue-axios'
9. import Snotify, { SnotifyPosition } from 'vue-snotify';
10. const options = {
11.   toast: {
12.     position: SnotifyPosition.rightBottom,
13.     showProgressBar: false,
14.     closeOnClick: false,
15.     pauseOnHover: true,
16.     timeout: 2000
17.   }
18. }
19.
20. Vue.use(Snotify, options)
21. Vue.use(VueRouter);
22. Vue.use(VueAxios, axios);
23. Vue.use(Vuetify);
24. Vue.config.productionTip = false;
25.
26. import Home from './components/Home'
27. import PatientIndex from './components/Patient/index'
28. import PatientCreate from './components/Patient/create'
29. import PatientShow from './components/Patient/show'
30. import ObservationIndex from './components/Observation/index'
31. import ObservationCreate from './components/Observation/create'
32. import PatientObservationCreate from
  './components/Patient/observation-create'
33.
34. const routes = [
35.   {path: "/",name:'home', component: Home},
36.   {path: "/patients",name:'patients', component: PatientIndex},
37.   {path: "/patient/create", name:'patient-create', component:
  PatientCreate},
38.   {path: "/patient/show", name: 'patient-show', component:
  PatientShow},
39.   {path: "/patient/observation/create", name: 'patient-
  observation-create', component: PatientObservationCreate},
40.   {path: "/observation/create", name: 'observation-create',
  component: ObservationCreate},
41.   {path: "/observations", name: 'observations', component:
  ObservationIndex},
42.   // {path: "/observation/create", name: 'observation-create',
  component: ObservationCreate}
43. ]
44. const router = new VueRouter({
45.   routes,
46.   mode: 'history'
47. });
```



```

154.         </v-flex>
155.     <v-flex
156.         md3
157.     >
158.         <v-text-field
159.             v-
160.                 model="descriptionB"
161.                 :rules="nameRules"
162.                 :counter="15"
163.                 label="SNOMED
164.                 description"
165.                 required
166.             >>/v-text-field>
167.     </v-flex>
168. </v-layout>
169. <v-layout>
170.     <v-flex
171.         xs12
172.         md3
173.     >
174.         <v-text-field
175.             v-model="codeS"
176.             :rules="nameRules"
177.             :counter="30"
178.             label="SNOMED Code
179.             Systolic BP"
180.             required
181.         >>/v-text-field>
182.     </v-flex>
183.     <v-flex
184.         xs12
185.         md3
186.     >
187.         <v-text-field
188.             v-
189.                 model="descriptionS"
190.                 label="Systolic BP"
191.                 required
192.             >>/v-text-field>
193.     </v-flex>
194.     <v-flex
195.         xs12
196.         md3
197.     >
198.         <v-text-field
199.             v-model="valueS"
200.             :rules="nameRules"
201.             :counter="15"
202.             label="Systolic
203.             Value"
204.             required
205.         >>/v-text-field>
206.     </v-flex>
207. </v-layout>
208. </v-layout>
209.     <v-flex
210.         xs12
211.         md3
212.     >

```

```

210.             <v-text-field
211.                 v-model="codeD"
212.                 :rules="nameRules"
213.                 :counter="30"
214.                 label="Code
    Diastolic"
215.                 required
216.             ></v-text-field>
217.         </v-flex>
218.     <v-flex
219.         xs12
220.         md3
221.     >
222.         <v-text-field
223.             v-
    model="descriptionD"
224.             label="SNOMED
    Description"
225.             required
226.         ></v-text-field>
227.     </v-flex>
228.     <v-flex
229.         xs12
230.         md3
231.     >
232.         <v-text-field
233.             v-model="valueD"
234.             :rules="nameRules"
235.             :counter="15"
236.             label="Diastolic
    Value"
237.             required
238.         ></v-text-field>
239.     </v-flex>
240. </v-layout>
241. <v-layout>
242.     <v-flex
243.         xs12
244.         md3
245.     >
246.         <v-btn color="primary"
    @click="saveBloodPreasure" block> Save</v-btn>
247.     </v-flex>
248. </v-layout>
249.
250. </v-container>
251. </v-form>
252. </v-card>
253.
254. </v-flex>
255. </v-layout>
256. </v-layout>
257. </v-container>
258. </template>
259.
260. <script>
261.     export default {
262.         data: () => ({
263.             baseUrl: 'http://localhost:8080/hapi-fhir-jpaserver-
    example/baseDstu3/',
264.             data: '',

```



```

321.         "text": "Blood pressure systolic &
    diastolic"
322.     },
323.     "subject": {
324.         "reference": "Patient/"+this.patient_id
325.     },
326.     "effectiveDateTime": this.date,
327.     "bodySite": {
328.         "coding": [
329.             {
330.                 "system": "http://snomed.info/sct",
331.                 "code": this.bodysite,
332.                 "display": this.descriptionB
333.             }
334.         ]
335.     },
336.     "component": [
337.         {
338.             "code": {
339.                 "coding": [
340.                     {
341.                         "system":
342.                             "http://snomed.info/sct",
343.                         "code": this.codeS,
344.                         "display":
345.                             this.descriptionS
346.                     }
347.                 ],
348.                 "valueQuantity": {
349.                     "value": this.valueS,
350.                     "unit": "mmHg",
351.                     "system":
352.                         "http://unitsofmeasure.org",
353.                     "code": "mm[Hg]"
354.                 },
355.                 "code": {
356.                     "coding": [
357.                         {
358.                             "system":
359.                                 "http://loinc.org",
360.                             "code": this.codeD,
361.                             "display":
362.                                 this.descriptionD
363.                         }
364.                     ],
365.                     "valueQuantity": {
366.                         "value": this.valueD,
367.                         "unit": "mmHg",
368.                         "system":
369.                             "http://unitsofmeasure.org",
370.                         "code": "mm[Hg]"
371.                     },
372.                 }
373.             ];
    this axios.post(this.baseUrl + 'Observation',
    data).then((response) => {

```

```

374.             console.log(response.data);
375.             this.$notify.success('Successfully added');
376.             this.$router.push({name:'observations'});
377.         });
378.     },
379.     getAllPatients() {
380.         this.axios.get(this.baseUrl +
381. 'Patient').then((response) => {
382.             console.log(response.data);
383.             this.patients = response.data.entry
384.         });
385.     },
386.     mounted() {
387.         this.getAllPatients();
388.     }
389. }
390. </script>
391.
392. <style>
393.
394. </style>
395.
396. <template>
397.   <v-container>
398.     <v-layout
399.       text-xs-center
400.       wrap
401.     >
402.       <v-flex sm3 justify-end>
403.         <v-btn color="info" block
404.           @click="$router.push({name:'observation-create'})">Add New
405.           Observation</v-btn>
406.         </v-flex>
407.       </v-layout>
408.       <v-layout class="py-10">
409.         <v-flex>
410.           <template v-for="observation in observations">
411.             <v-card>
412.               <v-card-title>Blood Pressure</v-card-title>
413.               <v-card-text>
414.                 <p><b>Systolic:</b>
415.                   {{observation.resource.component[0].valueQuantity.value}} <b>SNOMED
416.                   code for
417.                   systolic:</b>{{observation.resource.component[0].code.coding[0].code}
418.                 }</p>
419.                 <p><b>Diastolic:</b>
420.                   {{observation.resource.component[1].valueQuantity.value}} <b>SNOMED
421.                   code for
422.                   diastolic:</b>{{observation.resource.component[1].code.coding[0].code}
423.                 }</p>

```

```

424.   export default {
425.     data: () => ({
426.       baseUrl: 'http://localhost:8080/hapi-fhir-jpaserver-
example/baseDstu3/',
427.       observations:[],
428.       searchText:''
429.     }),
430.     mounted(){
431.       this.getPatient()
432.     },
433.     methods:{
434.       getPatient(){
435.         this.$axios.get(this.baseUrl+'Observation').then((response)
=> {
436.           console.log(response.data);
437.           this.observations = response.data.entry
438.         })
439.       }
440.     }
441.   }
442. </script>
443.
444. <style>
445.
446. </style>
447.
448. <template>
449.   <v-container>
450.     <v-layout justify-center
451.       text-xs-center
452.       wrap
453.     >
454.       <v-card>
455.         <v-card-title>
456.           Create New Patient
457.         </v-card-title>
458.         <v-form v-model="valid">
459.           <v-container>
460.             <v-layout>
461.               <v-flex
462.                 xs12
463.                 md3
464.               >
465.                 <v-select
466.                   v-model="mr"
467.                   :rules="nameRules"
468.                   :counter="10"
469.                   label="Mr/mrs"
470.                   :items="['Mr', 'Mrs']"
471.                   required
472.                 ></v-select>
473.             </v-flex>
474.             <v-flex
475.               xs12
476.               md3
477.             >
478.               <v-text-field
479.                 v-model="firstname"
480.                 :rules="nameRules"
481.                 :counter="10"
482.                 label="First Name"

```

```

483.             required
484.             ></v-text-field>
485.         </v-flex>
486.
487.         <v-flex
488.             xs12
489.             md3
490.         >
491.             <v-text-field
492.                 v-model="familyname"
493.                 :rules="nameRules"
494.                 :counter="10"
495.                 label="Family name"
496.                 required
497.             ></v-text-field>
498.         </v-flex>
499.         <v-flex
500.             xs12
501.             md3
502.         >
503.             <v-btn color="primary"
504.                 @click="savePatient"> Save</v-btn>
505.         </v-flex>
506.     </v-layout>
507. <v-layout>
508.     <v-flex
509.         xs12
510.         md3
511.     >
512.         <v-text-field
513.             v-model="mobile"
514.             :rules="nameRules"
515.             :counter="10"
516.             label="Mobile Number"
517.             required
518.         ></v-text-field>
519.     </v-flex>
520.     <v-flex
521.         xs12
522.         md3
523.     >
524.         <v-select
525.             v-model="gender"
526.             :rules="nameRules"
527.             :counter="10"
528.             label="Gender"
529.             :items="['male','female']"
530.             required
531.         ></v-select>
532.     </v-flex>
533.     <v-flex
534.         xs12
535.         md3
536.     >
537.         <v-menu
538.             v-model="menu2"
539.             :close-on-content-
540.                 click="false"
541.             :nudge-right="40"
542.             lazy

```

```

541.             transition="scale-
    transition"
542.             offset-y
543.             full-width
544.             min-width="290px"
545.         >
546.         <v-text-field
547.             slot="activator"
548.             v-model="date"
549.             label="Date of Birth"
550.             hint="Date of birth"
551.             prepend-icon="mdi-
    calendar"
552.             readonly
553.         ></v-text-field>
554.         <v-date-picker :min="todaydate"
    v-model="date"
555.             @input="menu2 =
    false"></v-date-picker>
556.             </v-menu>
557.         </v-flex>
558.         <v-flex
559.             xs12
560.             md3
561.         >
562.         <v-text-field
563.             v-model="address"
564.             :rules="nameRules"
565.             :counter="50"
566.             label="Address"
567.             required
568.         ></v-text-field>
569.         </v-flex>
570.     </v-layout>
571. </v-container>
572. </v-form>
573. </v-card>
574. </v-layout>
575. </v-container>
576. </template>
577.
578. <script>
579.     export default {
580.         data: () => ({
581.             baseUrl: 'http://localhost:8080/hapi-fhir-jpaserver-
    example/baseDstu3/',
582.             data: '',
583.             valid: false,
584.             firstname: '',
585.             familyname: '',
586.             mobile: '',
587.             mr: '',
588.             date: '',
589.             menu2: false,
590.             todaydate: "",
591.             gender: '',
592.             dob: '',
593.             address: '',
594.             nameRules: [
595.                 v => !!v || 'Name is required',

```

```

596.         v => v.length <= 50 || 'Name must be less than 10
characters'
597.     ],
598.     email: '',
599.     emailRules: [
600.         v => !!v || 'E-mail is required',
601.         v => /.+@.+/.test(v) || 'E-mail must be valid'
602.     ]
603. }),
604. methods: {
605.     savePatient() {
606.         let data = {
607.             "resourceType": "Patient",
608.             "id": "1",
609.             "identifier": [
610.                 {
611.                     "use": "usual",
612.                     "type": {
613.                         "coding": [
614.                             {
615.                                 "system":
"http://terminology.hl7.org/CodeSystem/v2-0203",
616.                                 "code": this.mr
617.                             }
618.                         ]
619.                     },
620.                     "system": "uia.no",
621.                     "value": "12345",
622.
623.                     "assigner": {
624.                         "display": "UIA Health"
625.                     }
626.                 }
627.             ],
628.             "active": true,
629.             "name": [
630.                 {
631.                     "use": "official",
632.                     "family": this.familyname,
633.                     "given": [
634.                         this.firstname
635.                     ]
636.                 }
637.             ],
638.             "telecom": [
639.                 {
640.                     "use": "home"
641.                 },
642.                 {
643.                     "system": "phone",
644.                     "value": this.mobile,
645.                 },
646.             ],
647.             "gender": this.gender,
648.             "birthDate": this.dob,
649.             "deceasedBoolean": false,
650.             "address": [
651.                 {
652.                     "use": "home",
653.                     "address1": this.address
654.                 }

```

```

655.
656.         }
657.     ],
658.     };
659.
660.     this.axios.post(this.baseUrl + 'Patient',
data).then((response) => {
661.         console.log(response.data);
662.         this.$notify.success("Successfully saved!")
663.         this.$router.push({name:'patients'});
664.
665.     })
666. }
667. }
668. }
669. </script>
670.
671. <style>
672.
673. </style>
674.
675. <template>
676.     <v-container>
677.         <v-layout wrap>
678.             <v-flex>
679.                 <v-card elevation="0">
680.                     <v-card-text>
681.                         <v-layout>
682.                             <v-flex sm10>
683.                                 <p>List of all the patients.</p>
684.                             </v-flex>
685.                             <v-flex sm2>
686.                                 <v-btn color="info"
@click="$router.push({name:'patient-create'})">Add Patient</v-btn>
687.                             </v-flex>
688.                         </v-layout>
689.
690.                     </v-card-text>
691.                 </v-card>
692.             </v-flex>
693.
694.         </v-layout>
695.         <v-layout class="my-2"
696.             text-xs-center
697.             wrap
698.         >
699.             <v-text-field
700.                 v-model="searchText"
701.                 @input="onSearch"
702.                 solo append-icon="search" hide-details single-
line placeholder="search for patient"></v-text-field>
703.         </v-layout>
704.
705.         <v-layout>
706.             <v-btn sm6 class="mx-2" block
@click="previous">Previous</v-btn>
707.             <v-btn sm6 class="mx-2" block @click="next">Next</v-
btn>
708.         </v-layout>
709.         <p class="light-blue--text mx-2">Showing
{{patients.length}} entries</p>

```

```

710.         <v-layout>
711.             <v-flex>
712.                 <template v-for="patient in patients">
713.                     <v-card v-if="patient.resource.name.length &&
714.                         patient.resource.name[0].given" class="my-2 py-2"
715.                         elevation="0"
716.                         @click="$router.push({name:'patient-
717.                             show',query:{patient_id:patient.resource.id}})">
718.                         <v-card-
719.                             title>{{patient.resource.name[0].given[0] +"
720.                                 "+patient.resource.name[0].family }}
721.                         </v-card-title>
722.                         </v-card>
723.                     </template>
724.                 </v-flex>
725.             </v-layout>
726.         </v-container>
727.     </template>
728. <script>
729.     export default {
730.         data: () => ({
731.             baseUrl: 'http://localhost:8080/hapi-fhir-jpaserver-
732.                 example/baseDstu3/',
733.             patients: [],
734.             searchText: '',
735.             offset: 0
736.         }),
737.         mounted() {
738.             this.getPatients()
739.         },
740.         methods: {
741.             onSearch() {
742.                 this.$axios.get(this.baseUrl + 'Patient?name=' +
743.                     this.searchText).then((response) => {
744.                         console.log(response.data);
745.                         this.patients = response.data.entry
746.                     }),
747.             getPatients() {
748.                 this.$axios.get(this.baseUrl +
749.                     'Patient?_count=10&&_getpagesoffset=' + this.offset).then((response)
750.                     => {
751.                         console.log(response.data);
752.                         this.patients = response.data.entry
753.                     })
754.             },
755.             previous() {
756.                 if (this.offset > 0) {
757.                     this.offset -= 10;
758.                     this.getPatients()
759.                 }
760.             },
761.             next() {
762.                 this.offset += 10;
763.                 this.getPatients()
764.             }
765.         }
766.     }
767. </script>
768.

```



```

763. <style>
764.
765. </style>
766.
767. <template>
768.   <v-container>
769.     <!--<v-card>-->
770.     <!--<v-card-text>-->
771.       <!--<v-btn>Add Patient</v-btn>-->
772.     <!--</v-card-text>-->
773.   <!--</v-card>-->
774.   <v-layout>
775.     <v-flex sm3 justify-end>
776.       <!--<a href="" color="info" block
@click.prevent="$router.push({name:'patient-show'})">Go Back</a>-->
777.     </v-flex>
778.   </v-layout>
779.   <v-layout justify-center
780.     text-xs-center
781.     wrap
782.   >
783.
784.     <v-layout>
785.       <v-flex sm12>
786.         <v-card>
787.           <v-card-title>
788.             Create New Vital Sign Observation for
Blood Pressure
789.           </v-card-title>
790.           <v-form v-model="valid">
791.             <v-container>
792.               <v-layout v-if="patient.resource">
793.                 <v-flex
794.                   xs12
795.                   md3
796.                 >
797.                   <v-text-field
798.                     label="Patient
Name"
799.                     required
800.                     readonly
801.                     disabled
802.                     :label="patient.resource.name[0].given[0]"
803.                   ></v-text-field>
804.                 </v-flex>
805.                 <v-flex
806.                   xs12
807.                   md3
808.                 >
809.                   <v-menu
810.                     v-model="menu2"
811.                     :close-on-content-
click="false"
812.                     :nudge-right="40"
813.                     lazy
814.                     transition="scale-
transition"
815.                     offset-y
816.                     full-width
817.                     min-width="290px"

```

```

818.                >
819.                <v-text-field
820.                slot="activator"
821.                v-model="date"
822.                label="Date"
823.                hint="Departure"
824.                prepend-
825.                icon="mdi-calendar"
826.                readonly
827.                ></v-text-field>
828.                <v-date-picker
829.                :min="todaydate" v-model="date"
830.                @input="menu2 = false"></v-date-picker>
831.                </v-menu>
832.            </v-flex>
833.        </v-layout>
834.    <v-layout>
835.        <v-flex
836.            md3
837.            >
838.                <v-text-field
839.                v-model="bodysite"
840.                :rules="nameRules"
841.                :counter="15"
842.                label="SNOMED code
843.                for BodySite"
844.                required
845.                ></v-text-field>
846.            </v-flex>
847.        <v-flex
848.            md3
849.            >
850.                <v-text-field
851.                v-
852.                model="descriptionB"
853.                :rules="nameRules"
854.                :counter="15"
855.                label="SNOMED
856.                description"
857.                required
858.                ></v-text-field>
859.            </v-flex>
860.        </v-layout>
861.    <v-layout>
862.        <v-flex
863.            xs12
864.            md3
865.            >
866.                <v-text-field
867.                v-model="codeS"
868.                :rules="nameRules"
869.                :counter="30"
870.                label="SNOMED Code
871.                Systolic BP"
872.                required
873.                ></v-text-field>

```

```

870.         </v-flex>
871.     <v-flex
872.         xs12
873.         md3
874.     >
875.         <v-text-field
876.             v-
            model="descriptionS"
877.             label="SNOMED
            description Systolic BP"
878.             required
879.         >></v-text-field>
880.     </v-flex>
881.     <v-flex
882.         xs12
883.         md3
884.     >
885.         <v-text-field
886.             v-model="valueS"
887.             :rules="nameRules"
888.             :counter="15"
889.             label="Systolic
            Value"
890.             required
891.         >></v-text-field>
892.     </v-flex>
893. </v-layout>
894. <v-layout>
895.
896.     <v-flex
897.         xs12
898.         md3
899.     >
900.         <v-text-field
901.             v-model="codeD"
902.             :rules="nameRules"
903.             :counter="30"
904.             label="Code
            Diastolic"
905.             required
906.         >></v-text-field>
907.     </v-flex>
908.     <v-flex
909.         xs12
910.         md3
911.     >
912.         <v-text-field
913.             v-
            model="descriptionD"
914.             label="SNOMED
            Description"
915.             required
916.         >></v-text-field>
917.     </v-flex>
918.     <v-flex
919.         xs12
920.         md3
921.     >
922.         <v-text-field
923.             v-model="valueD"
924.             :rules="nameRules"

```

```

925.                                     :counter="15"
926.                                     label="Diastolic
    Value"
927.                                     required
928.                                     ></v-text-field>
929.                                 </v-flex>
930.                             </v-layout>
931.                         <v-layout>
932.                             <v-flex
933.                                 xs12
934.                                 md3
935.                             >
936.                                 <v-btn color="primary"
    @click="saveBloodPreasure" block> Save</v-btn>
937.                                 </v-flex>
938.                             </v-layout>
939.
940.                         </v-container>
941.                     </v-form>
942.                 </v-card>
943.
944.                 </v-flex>
945.             </v-layout>
946.         </v-layout>
947.     </v-container>
948. </template>
949.
950. <script>
951.     export default {
952.         data: () => ({
953.             baseUrl: 'http://localhost:8080/hapi-fhir-jpaserver-
    example/baseDstu3/',
954.             data: '',
955.             valid: false,
956.             patients:[],
957.             patient:{},
958.             patient_id: '',
959.             patientName:'',
960.             patient_reference: '',
961.             date: '',
962.             menu2: false,
963.             bodysite: '368209003',
964.             descriptionB: 'Right arm',
965.             codeS: '271649006',
966.             descriptionS: 'Systolic blood pressure',
967.             valueS: '',
968.             codeD: '271650006',
969.             descriptionD: 'Diastolic blood pressure',
970.             todaydate: new Date().toISOString().slice(0, 10),
971.             valueD: '',
972.             nameRules: [
973.                 v => !!v || 'Name is required',
974.                 v => v.length <= 30 || 'Name must be less than 10
    characters'
975.             ],
976.             email: '',
977.             emailRules: [
978.                 v => !!v || 'E-mail is required',
979.                 v => /.+@.+/.test(v) || 'E-mail must be valid'
980.             ]
981.         })

```

```

982.         methods: {
983.             saveBloodPreasure() {
984.                 let data = {
985.                     "resourceType": "Observation",
986.                     "id": "blood-pressure",
987.                     "meta": {
988.                         "profile": [
989.                             "http://hl7.org/fhir/StructureDefinition/vitalsigns"
990.                         ]
991.                     },
992.
993.                     "status": "final",
994.                     "category": [
995.                         {
996.                             "coding": [
997.                                 {
998.                                     "system":
1000.                                     "http://terminology.hl7.org/CodeSystem/observation-category",
1001.                                     "code": "vital-signs",
1002.                                     "display": "Vital
1003.                                     Signs"
1004.                                 }
1005.                             ]
1006.                         },
1007.                         {
1008.                             "code": {
1009.                                 "coding": [
1010.                                     {
1011.                                         "system":
1012.                                         "http://snomed.info/sct",
1013.                                         "code": "75367002",
1014.                                         "display": "Blood pressure
1015.                                         panel with all children optional"
1016.                                     }
1017.                                 ],
1018.                                 "text": "Blood pressure systolic &
1019.                                 diastolic"
1020.                             },
1021.                             "subject": {
1022.                                 "reference":
1023.                                 "Patient/"+this.patient_id
1024.                             },
1025.                             "effectiveDateTime": this.date,
1026.                             "bodySite": {
1027.                                 "coding": [
1028.                                     {
1029.                                         "system":
1030.                                         "http://snomed.info/sct",
1031.                                         "code": this.bodysite,
1032.                                         "display":
1033.                                         this.descriptionB
1034.                                     }
1035.                                 ]
1036.                             },
1037.                             "component": [
1038.                                 {
1039.                                     "code": {
1040.                                         "coding": [
1041.                                             {

```

```

1033.                                     "system":
      "http://snomed.info/sct",
1034.                                     "code": this.codeS,
1035.                                     "display":
      this.descriptionS
1036.                                     }
1037.                                     ]
1038.                                     },
1039.                                     "valueQuantity": {
1040.                                       "value": this.valueS,
1041.                                       "unit": "mmHg",
1042.                                       "system":
      "http://unitsofmeasure.org",
1043.                                       "code": "mm[Hg]"
1044.                                     },
1045.                                     },
1046.                                     {
1047.                                       "code": {
1048.                                         "coding": [
1049.                                           {
1050.                                             "system":
      "http://snomed.info/sct",
1051.                                             "code": this.codeD,
1052.                                             "display":
      this.descriptionD
1053.                                           }
1054.                                         ]
1055.                                       },
1056.                                       "valueQuantity": {
1057.                                         "value": this.valueD,
1058.                                         "unit": "mmHg",
1059.                                         "system":
      "http://unitsofmeasure.org",
1060.                                         "code": "mm[Hg]"
1061.                                       },
1062.                                     }
1063.                                     ]
1064.                                     };
1065.                                     this.axios.post(this.baseUrl +
      'Observation', data).then((response) => {
1066.                                       console.log(response.data);
1067.                                       this.$notify.success('Successfully
      added');
1068.                                       this.$router.push({name:'patient-
      show'});
1069.                                       });
1070.                                     },
1071.                                     getPatientById() {
1072.                                       this.axios.get(this.baseUrl +
      'Patient?_id=' + this.$route.query.patient_id).then((response) => {
1073.                                         console.log(response.data);
1074.                                         this.patient = response.data.entry[0]
1075.                                       })
1076.                                     },
1077.                                     },
1078.                                     mounted() {
1079.                                       this.patient_id=this.$route.query.patient_id;
1080.                                       this.getPatientById();
1081.                                     }
1082.                                     }
1083.                                     </script>

```

```

1084.
1085.     <style>
1086.
1087.     </style>
1088.
1089.     <template>
1090.         <v-container>
1091.
1092.             <v-card elevation="0">
1093.                 <v-card-text>
1094.                     <v-layout>
1095.                         <v-flex sm10>
1096.                             <h3 v-if="
1097. patient[0]">{{patient[0].resource.name[0].given[0]+"
1098. "+patient[0].resource.name[0].family}} Observations</h3>
1099.
1100.                         </v-flex>
1101.                         <v-flex sm2>
1102.                             <v-btn color="info" v-
1103. if="patient[0]"
1104. @click="$router.push({name:'patient-observation-
1105. create',query:{patient_id:patient[0].resource.id}})">
1106.                                 New Observation
1107.                             </v-btn>
1108.                         </v-flex>
1109.                     </v-layout>
1110.                 </v-card-text>
1111.             </v-card>
1112.
1113.         <v-layout
1114.             wrap
1115.         >
1116.             <v-flex>
1117.                 <template v-for="observation in
1118. observations">
1119.                     <v-card v-
1120. if="observation.resource.component" class="my-2" elevation="0">
1121.                         <v-card-title>Blood Pressure</v-
1122. card-title>
1123.                         <v-card-text>
1124.                             <p><b>{{observation.resource.component[0].code.coding[0].display}}:</
1125. b>
1126.                             {{observation.resource.component[0].valueQuantity.value}} <b>Snomed
1127. code for
1128.                             systolic:</b>{{observation.resource.component[0].code.coding[0].code}
1129. }}</p>
1130.                             <p><b>{{observation.resource.component[1].code.coding[0].display}}:</
1131. b>
1132.                             {{observation.resource.component[1].valueQuantity.value}} <b>Snomed
1133. code for
1134.                             diastolic:</b>{{observation.resource.component[1].code.coding[0].code
1135. }}</p>
1136.                         </v-card-text>

```

```

1125.             </v-card>
1126.         </template>
1127.
1128.     </v-flex>
1129.
1130.     </v-layout>
1131. </v-container>
1132. </template>
1133.
1134. <script>
1135.     export default {
1136.       data: () => ({
1137.         baseUrl: 'http://localhost:8080/hapi-fhir-
1138.           jpaserver-example/baseDstu3/',
1139.         patients: [],
1140.         observations: {},
1141.         searchText: '',
1142.         patient: {}
1143.       }),
1144.       mounted() {
1145.         this.getPatientObservations()
1146.         this.getPatientById()
1147.       },
1148.       methods: {
1149.         getPatientObservations() {
1150.           this.axios.get(this.baseUrl +
1151.             'Observation?subject:Patient=' +
1152.             this.$route.query.patient_id).then((response) => {
1153.             console.log(response.data);
1154.             this.observations = response.data.entry
1155.           })
1156.         },
1157.         getPatientById() {
1158.           this.axios.get(this.baseUrl +
1159.             'Patient?_id=' + this.$route.query.patient_id).then((response) => {
1160.             console.log(response.data);
1161.             this.patient = response.data.entry
1162.           })
1163.         },
1164.       }
1165.     }
1166. </script>
1167.
1168. <style>
1169. </style>
1170.
1171. <template>
1172.   <v-container>
1173.     <v-card elevation="0">
1174.       <v-card-text>
1175.         <v-layout
1176.           text-xs-center
1177.           wrap
1178.         >
1179.           <v-btn class="mx-2" color="info" block
1180.             @click="$router.push({name: 'patients'})">View Patients</v-btn>
1181.           <v-btn class="mx-2" color="info" block
1182.             @click="$router.push({name: 'observations'})">View Observations</v-
1183.             btn>
1184.         </v-layout>

```



```
1179.         </v-card-text>
1180.     </v-card>
1181.
1182.     </v-container>
1183. </template>
1184.
1185. <script>
1186.     export default {
1187.         data: () => ({
1188.
1189.             })
1190.     }
1191. </script>
1192.
1193. <style>
1194.
1195. </style>
```