TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Lilia Tünts 211857IAPM

ISIKLIKE TERVISEANDMETE DETSENTRALISEERITUD SISU-ADRESSEERITAVAS SALVESTUSVÕRGUS HOIDMISE JA KASUTAMISE KONTSEPTSIOONI VALIDEERIMINE

Magistritöö

Juhendaja: Gunnar Piho
PhD

Kaasjuhendaja: Toomas Klementi
MSc

Tallinn 2025

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Author: Lilia Tünts

12.05.2025

Annotatsioon

Käesoleva lõputöö eesmärgiks on valideerida isiklike terviseandmete detsentraliseeritud sisu-adresseeritavas salvestusvõrgus hoidmise ja kasutamise kontseptsiooni. Lisaks, uurida tehnilise teostuse võimalusi kasutada isiklike terviseandmete omamisel, säilitamisel ja kasutamisel detsentraliseeritud sisu-adresseeritavaid salvestusvõrke. Esitatud kontseptsiooni tehnilise teostatavuse valideerimiseks luua rakenduse prototüüp (PoC) rakenduse valmidustasemel TRL 3.

Tänapäeval paiknevad inimeste terviseandmed peamiselt haiglate, perearstide või riiklike ja erialapõhiste terviseinfosüsteemide hallatavates serverites. Käesolevas töös keskendutakse võimalustele talletada terviseandmeid viisil, mis annab inimesele täieliku kontrolli oma terviseandmete üle. Uurimistöö lähtub TalTechi eMedLabi uurimisgrupi tegevusest, kus käsitletakse detsentraliseeritud sisu-adresseeritavate salvestusvõrkude rakendatavust terviseandmete turvalisel hoiustamisel.

Magistritöö tulemusena valmisid potentsiaalse uue süsteemi esialgne analüüs ja arhitektuuri tutvustamine ning idee tehnilise teostatavuse valideerimiseks prototüüp-rakendus. Samuti olid läbi mõeldud tuleviku väljakutsed, detailselt uuritud detsentraliseeritud sisu-adresseeritavate salvestusvõrkude omadused ja tehnilise teostatavuse takistused ja võimalused.

Lõputöös püstitatud eesmärk oli täidetud. Tulemuste põhjal kirjeldas autor edasised uurimissuunad ning järgmised sammud, mida edasises töös ette võtta. Samuti sõnastati võimalikud uurimisteemad ja suunad, mida saab käsitleda tulevases doktoritöös.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 61 leheküljel, 5 peatükki, 26 joonist, 5 tabelit.

Abstract

Validation of the Concept of Storing and Using Personal Health Records in a Decentralised Content-Addressable Storage Network

The objective of this master's thesis is to validate the concept of storing and using personal health data in a decentralised content-addressable storage network. Additionally, it explores the technical implementation possibilities for owning, storing, and using personal health data through described decentralised networks. To validate the technical feasibility of the proposed concept, a proof-of-concept (PoC) application prototype was developed at technology readiness level (TRL) 3.

Currently, individuals' health data or PHR are primarily stored on servers managed by hospitals, general practitioners, or national and domain-specific health information systems. This thesis focuses on exploring solutions that would enable individuals to have full control over their own health data. The research is based on the work of the TalTech eMedLab research group, which investigates the applicability of decentralised content-addressable storage networks for the secure storage of health data.

As a result of the thesis, an initial analysis and architectural overview of a potential new system were developed, along with a prototype application to validate the technical feasibility of the idea. Future challenges were also considered, and the characteristics, obstacles, and opportunities of decentralised content-addressable storage networks were studied in detail.

The objective of the thesis was achieved. Based on the results, the author outlined further research directions and the next steps to be taken in future work. Potential research topics and areas were also identified for possible exploration in a future PhD thesis.

The thesis is written in estonian language and is 61 pages long, including 5 chapters, 26 figures and 5 tables.

Lühendite ja mõistete sõnastik

PoC	*Proof of Concept*, kontseptsiooni valideerimine [1]
SPF	*Single Point of Failure*, osa süsteemis, mille rikke võib põhjustada terve süsteemi katkestust
CPU	*Central Processing Unit*, arvutiprotsessor
P2P	*Peer-to-peer*, võrdõigusvõrk, võrdsete õiguste ja võimalustega võrgusõlmede kogum
FHIR	*Fast Healthcare Interoperability Resources*, standard, reeglite ja spetsifikatsioonide kogum meditsiiniandmete vahetamiseks elektroonilisel kujul. Standardi paindlikkus ja kohandatavus võimaldab selle kasutamise erinevate tervishoiu infosüsteemidega [2]
HL7	*Health Level Seven*. Tegemist on OSI (Open Systems Interconnection [3]) järgi kõrgema (seitsmenda, rakenduse) taseme protokolliga andmete vahetamiseks tervishoiuga seotud süsteemides. Sisalab hulga ülemaailmseid standardeid kliiniliste ja administratiivsete terviseandmete edastamiseks rakenduste vahel eesmärgiga parandada tervishoiusüsteemide toimivust ja patsientide heaolu [4]
JSON	*JavaScript Object Notation*, andmevahetuse vorming
UI	*User Interface*, kasutajaliides
BZZ	Etherium Swarm detsetntraliseeritud sisu-addresseeritavas salvestusvõrgus kasutatav elektrooniline maksevahend [5]
DAI	Etherium Swarm detsetntraliseeritud sisu-addresseeritavas salvestusvõrgus kasutatav elektroonilise maksevahendi stabiilne krüptoraha kursiga 1$ = 1 DAI [6]
TRL	*Technology Readiness Level*, Esialgselt NASA poolt välja töötatud tehnoloogia küpsustasemete skaale selleks, et hinnata tehnoloogia valmisolekut reaalseks kasutuseks [7]
DICOM	*Digital Imaging and Communications in Medicine*, meditsiini piltide ja sellega seotud informatsiooni kuvamise ja jagamise rahvusvaheline formaat [8]
PHR	*Personal Health Record*, isiklikud terviseandmed

EHDS	*European Health Data Space*, on Euroopa Komisjoni algatus (ja kavandatav õigusraamistik), mille eesmärk on luua ühtne, turvaline ning läbipaistev digitaalne infrastruktuur terviseandmete jagamiseks ja kasutamiseks kogu Euroopa Liidus
DCAS	*Decentralised Content-Addressable Storage Network*, detsentraliseeritud sisu-addresseeritav salvestusvõrk, kus andmeid otsitakse ja tuvastatakse nende sisu (näiteks krüptograafilise räsi/hashi) alusel, mitte traditsiooniliselt serveri või asukoha (URL-i või IP-aadressi) järgi
GDPR	*General Data Protection Regulation*, Euroopa Liidu määrus, mis kaitseb isikuandmete privaatsust
DSR	*Design Science Research*, on uurimismeetod, mida kasutatakse eelkõige infotehnoloogia, inseneriteaduste ja ärijuhtimise valdkonnas, et luua ja hinnata uuenduslikke lahendusi (artefakte) reaalse maailma probleemide lahendamiseks [9]
Web3	*World Wide Web 3.0*, maailma veebi ehk võrgustikku uus iteratsioon, mis hõlmab sellised kontseptsioonid nagu detsentraliseerimine, plokiahela tehnoloogiad ning token'itel põhinev majandus [10]
eMedLab	Tallinna Tehnikaülikooli (TalTech) digitaaltervise ja meditsiiniinformaatika uurimisrühm, mis ühendab mitteametlikult osasid IT-teaduskonna E-tervise keskuse ja Äriinfotehnoloogia uurimisrühmade teadlasi
Põhivõti, peamine võti	DCAS süsteemis andmete identifitseerimiseks andmete sisust arvutatud räsiväärtus
TTL	*Time To Live*, andmete säilimise maksimaalse perioodi väärtus täisarvuna

Sisukord

Jooniste loetelu

Tabelite loetelu

1. Sissejuhatus

Praegu on inimese terviseandmed kas haiglate, perearstide või riiklike ja harukondlike teriviseinfosüsteemide hallatavates serverites. Selliste suurte keskselt hallatavate süsteemide opereerimine ja nendes turvalisuse tagamine on tervishoiusüsteemile väga kallis. Näiteks, 2020. aastal moodustasid tervishoiukulud Eesti riigi sisemajanduse koguproduktist 7,8% [11]. Samas on igal aastal maailmas suuri andmeründe id [12], mille tagajärjel paljude inimeste privaatsed terviseandmed on lekkinud. TalTech eMedLab uurimisgrupis uuritakse võimalusi terviseandmete hoidmisel inimese omanduses ja täieliku kontrolli all olevates detsentraliseeritud sisu-adresseeritavates salvestusvõrkudes.

Lõputöö raames uuritakse isiklike terviseandmete detsentraliseeritud sisu-adresseeritavates salvestusvõrkudes hoidmise ja kasutamise tehnilist teostatavust.

Terviseandmete hoidmine detsentraliseeritud sisu-adresseeritavas salvestusvõrgus muudab oluliselt traditsioonilisi andmete kuuluvuse, jagamise ja omandamise põhimõtteid. Traditsioonilistes süsteemides kuuluvad andmed tervishoiuasutustele või pilveteenuse pakkujatele, kes haldavad nende turvalisust ja ligipääsuõigusi. Detsentraliseeritud sisu-adresseeritavas salvestusvõrgus on andmed salvestatud hajutatud sõlmede vahel ning nende haldamine toimub kasutaja kontrolli all. See tähendab, et andmete omanikuks jääb kasutaja ise ning ta saab määrata, kellele ja millistel tingimustel juurdepääs võimaldatakse. Lisaks, väheneb andmelekkete ja loata juurdepääsu risk, kuna andmed ei asu ühes kindlas serveris, vaid on hajutatud üle võrgu.

Käesolevas lõputöös uuritakse tehnilise teostuse võimalusi kasutada isiklike terviseandmete omamisel, säilitamisel ja kasutamisel detsentraliseeritud sisu-adresseeritavaid salvestusvõrke. Töö raames katsetati Ethereum Swarm [13] detsentraliseeritud sisu-adresseeritava salvestusvõrgu kasutamise, selle integreerimise ja rakendamise võimalusi isiklike terviseandmete hoidmiseks ja kasutamiseks. Katsetamiseks arendati rakenduse kontseptsiooni valideerimise (PoC) prototüüp [1]. Prototüüp demonstreerib isiklike terviseandmetega opereerimise võimalusi detsentraliseeritud sisu-adresseeritava salvestusvõrgu abil.

Uuringute käigus antud teema oli vaadeldud ja käsitletud mitmest vaatenurgast: tehnilised lahendused, eetilised küsimused ning ühiskondlikud aspektid. Kontseptsiooni valideerimiseks arendatakse rakendus, millega saab pakutud kontseptsiooni testida lihtsustatud kujul

ning vastavalt tehnoloogia valmidusastmele TRL3 [7].

1.1 Probleem

T. Klementi on sõnastanud peamised isiklike terviseandmetega seotud probleemid kolme dilemmana [14].

Esimene dilemma on isiklike terviseandmete ligipääsetavus. Terviseandmed on tundlikud isiklikud andmed ning peavad olema hästi kaitstud. Samal ajal, isiklikud terviseandmed omavad suurt väärtust teaduses ja uute tehnoloogiate arengus. Seetõttu üritatakse saavutada samal ajal nii andmete kaitstust, andmete ligipääsetavust ja kättesaadavust ning terviseandmete omaniku anonüümsuse säilitamist.

Teine dilemma on isiklike terviseandmete kogumine ja hoidmine. Andmed on hoitud laiali erinevates meditsiiniasutustes, osaliselt isiklikes seadmetes ja erinevatel andmekandjatel (nii digitaalsel, kui ka paberikandja kujul). On oluline saada ühe inimese kõik terviseandmed ühte kohta sellepärast, et puuduolevad terviseandmete osad võivad rikkuda terviklikku pildi patsiendi tervisest. Puuduolevad andmete tükid võivad mõjutada patsiendi tervise suhtes tehtavaid otsuseid, järeldusi ning ravi tulemusi.

Samal ajal on oluline, et ühes kohas oleksid hoitud ainult ühe inimese andmed. Sellisel juhul rünnata saab korraga ainult ühe inimese andmeid, ilma teisi inimesi andmeid puudutamata. Järelikult, küberrünnaku tulemusena võimalik saadav kasu on väga väike võrreldes küberrünnaku enda hinnaga. Üks hiljutine näide Eestis juhtunud andmelekkest (ligi 700 000 isikukoodi) juhtus Apotheka kliendikaartide omanike andmetega [15]. Terviseandmete andmelekke maailma statistika järgi küberrünnete arv terviseandmete suhtes on igaaastaselt kasvanud suuremaks, ning selle arvuga koos kasvab ka kannatanud isikute arv [12].

Kolmas dilemma on isiklike terviseandmete omandiõigus. Vastavalt GDPR [16] seadusele ja EHDS [17] visioonile, inimesel peab olema ülevaade enda andmete kasutamisest. Kuna andmed aga asuvad kolmandate osapoolte (haiglad, riiklikud süsteemid, muud meditsiini- ja terviseasutused) serverites, siis inimesel on tegelikkuses väga vähe proaktiivseid ja ennetavaid võimalusi kontrollida enda andmete kasutamist.

Üks võimalus on terviseandmete hoiustamisel inimese omandiõiguse ja täieliku kontrolli all olevates detsentraliseeritud sisu-adresseeritavas salvestusvõrgus. [14].

1.2 Eesmärk

Magistritöö eesmärk on analüüsida, kavandada kasutusjuhud ja arendada API-d selleks, et valideerida detsentraliseeritud sisu-adresseeritavas salvestusvõrgus terviseandmete hoidmise ja kasutamise tehnilist teostatavust ja realiseerida TRL 3 tasemel [7] prototüüp selleks, et saaks hakata valideerima tehnoloogia kasutatavust kõrgematel TRL tasemetel.

Püstitatud eesmärgi edukaks saavutamiseks on vaja:

1. Uurida, mis on sobiv raamistik ja *tech stack* tehnilise lahenduse teostamiseks, et ühe koodibaasi põhjal teha Android, iOS, Windows ja MacOS operatsioonsüsteemidele rakendused.
2. Uurida, kas veebipõhine lahendus sobiks ettenähtud lahenduse teostamiseks.
3. Arendada rakendust võimalikult üldisel kujul, et võimaldada kasutada terviseandmed erinevates FHIR formaatides - JSON, FHIR Shorthand, RDF Turtle.
4. Arendada terviseandmete, k.a. DICOM piltide salvestamist ja kuvamist FHIR ressursside kujul.

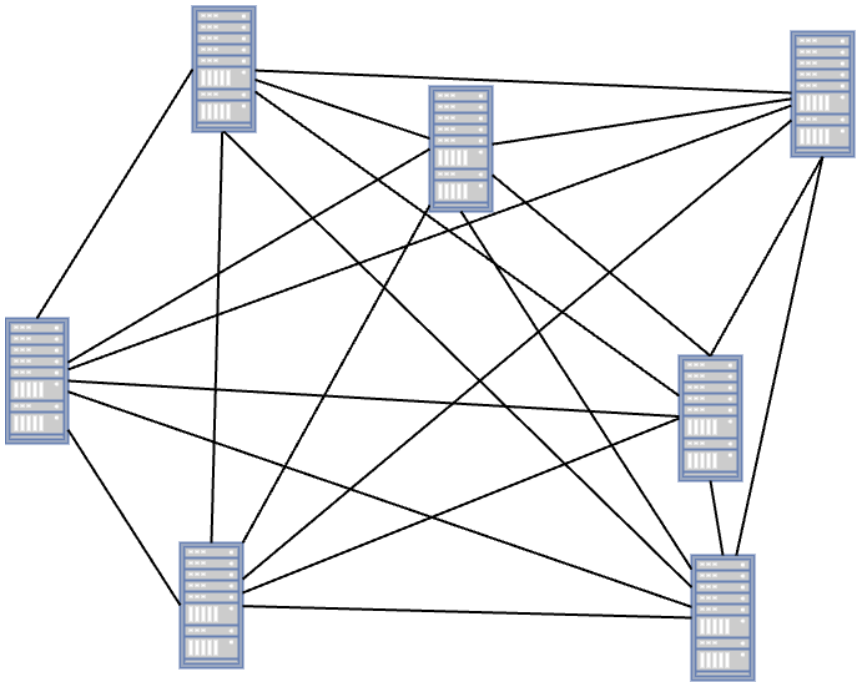
2. Metoodika

2.1 Uurimisobjekt

Käesoleva lõputöö uurimisobjektiks on MAUI, .NET Core ja Ethereum Swarm vahenditega tehnilise teostatavuse esialgne valideerimine, et võimalikult varakult leida potentsiaalselt probleemsed kohad.

Detsentraliseeritud süsteem

Detsentraliseeritud süsteemi põhimõte seisneb selles, et süsteemil puudub keskvõim. Süsteemi moodustavad autonoomsed osalejad, mis suhtlevad omavahel ning kujundavad sellisel viisil terviklikku süsteemi [18]. Süsteemi käitumine määratakse teatud algoritmiga.



Joonis 1. *Detsentraliseeritud süsteem*

2.2 Ligipääs andmetele DCAS võrgus

Detsentraliseeritud sisu-adresseeritava salvestusvõrgu toimimise põhimõtted

Detsentraliseeritud sisu-adresseeritav salvestusvõrk (DCAS) on andmete hoidmise ja jagamise viis, mis erineb oluliselt traditsioonilistest tsentraliseeritud andmebaasidest.

Detsentraliseeritud sisu-adresseeritavas salvestusvõrgus suure hulga andmed salvestatakse väga väikeste andmeosadena suures hulgas masinates laiali. Andmeosad salvestatakse nii:

- väikesed andmehulgad ei ole eraldiseisvalt loetavad,
- ükski masin, mille peale said andmete tükid salvestatud, ei tea mis on salvestatud andmete tükki allikas,
- ükski masin, mille peale said andmete tükid salvestatud, ei tea millised andmed hoitakse

Andmete identifitseerimine toimub mitte andmete füüsilise aadressi järgi, vaid andmete sisu järgi.

Andmete jagamine ja adresseerimine

Detsentraliseeritud sisu-adresseeritava salvestusvõrgu andmete sisestamisel tükeldatakse andmed väiksemateks osadeks (*shard*'ideks). Andmete tükid salvestatakse hajutatud sõlmedes üle kogu võrgu. Andmete tükeldamine ja hajutatud sõlmede peale salvestamine tagab, et kui osa sõlmedest hävitatakse, jäävad andmed kättesaadavaks teistes sõlmedes.

Traditsioonilistes andmete salvestamise meetodites kasutatakse aadresse (nt serveri URL), et saada ligipääsu tervele andmete komplektile. Ilma tervele andmekogusele ligipääsu saamata, ei ole võimalik üksiku andmete tükki kättesaamine. Detsentraliseeritud sisu-adresseeritavas võrgus identifitseeritakse andmed nende räsiväärtuse (*cryptographic hash*) kaudu, mis arvutatakse andmete sisust. See tähendab, et sama sisu annab alati sama identifikaatori, muutes manipuleerimise keeruliseks. Sisu muutmisel muutub ka andmete ligipääsu aadress ning andmete tükki käsitletakse edaspidi kui uut.

Andmete muutmine

Detsentraliseeritud sisu-adresseeritavas salvestusvõrgus on andmete muutmine palju keerulisem, kuna DCAS süsteemides andmete identifitseerimine põhineb sisu räsiväärtustel. Iga salvestatud andmeühik saab unikaalse räsiväärtuse, mis sõltub täielikult selle sisu konkreetsest väärtusest. Kui andmete tükk muudetakse, muutub ka selle tükki räsiväärtus. Selle tõttu käsitletakse muudatusi kui täiesti uusi andmeversioone. See tähendab, et:

1. originaalandmeid ei saa otse muuta. Andmete aadress sõltub andmete sisust, isegi väike muudatus muudab räsiväärtuse ning süsteem loob uue andmeploki;
2. versioonihaldus lahendatakse teistmoodi. Uut versiooni käsitletakse eraldiseisva andmeobjektina. Seetõttu peab olema eraldi hoitud viide praegusel hetkel kehtiva

versioonile andmetest.

See erinevus muudab andmete hoidmist detsentraliseeritud sisu-adresseeritavas salvestusvõrgus tugevamaks andmete manipuleerimise ja volitamata muutmise vastu.

Võrdlus traditsiooniliste andmebaasidega

Tabel 1. Detsentraliseeritud sisu-adresseeritava salvestusvõrgu võrdlus traditsiooniliste andmebaasidega

Omadus	Andmebaas	DCAS
Andmete asukoht	Määratud server või pilveteenus	Väikeste osadena hajutatult suures hulgas detsentraliseeritud võrgu sõlmedes
Adresseerimine	Füüsiline aadress (server, IP)	Sisu-põhine räsiväärtus
Ligipääsu kontroll	Tsentraliseeritud autentimine	Krüptograafilised võtmed
Turvalisus	Serveri või teenuse haldaja määratud	Krüptograafia ja võtmepõhine juurdepääs
Kättesaadavus	Sõltub andmebaasi kättesaadavusest	Hajutatud sõlmed tagavad kättesaadavuse
Andmete muutmine	Võimalik administratiivsete õigustega	Muutmine keeruline, kuna muudatus tähendab uue andmete komplekti loomist ning järelikult ka ligipääsu võtme väärtuse muutmist

2.3 Töös kasutatud tööriistad

Töö tehnilise osa teostamiseks autor otsustas luua rakendust, mis töötaks erinevatel platvormidel. Rakendus on arendatud selleks, et katsetada ja kontrollida .NET MAUI raamistiku [19] võimalusi detsentraliseeritud sisu-adresseeritavas salvestusvõrguga integreerimiseks selle eesmärgiga, et luua vastavad API-d terviseandmete hoidmiseks ja kasutamiseks. Arenduse töös oli kasutatud JetBrains Rider arenduskeskkond. Koodi hoidmiseks ja jagamiseks oli kasutatud GitLab versioonihaldussüsteem.

Ethereum Swarm

Detsentraliseeritud sisu-adresseeritava salvestusvõrguks oli valitud Ethereum Swarm võrk [13]. Swarm oli valitud detsentraliseeritud sisu-adresseeritava salvestusvõrguna käesoleva töö raames sellepärast, et:

1. Swarm on jõudnud tehnoloogilise valmiduse tasemele TRL 8 [7], mis tähendab, et:
   - süsteem on integreeritud ja demonstreeritud reaalses keskkonnas;

    - on olemas juba toimiv ökosüsteem ja aktiivne kasutajaskond.
2. Swarm on avatud lähtekoodiga *(open-source)* projekt:
    - kogu kood on avalikult GitHub'is [20] saadaval;
    - see võimaldab läbipaistvust, sõltumatut auditeerimist ja arenduse kogukonna tuge;
    - see vähendab oluliselt riske, mis võivad olla seotud nn *black-box* ehk suletud ja teadmata funktsionaalsusega tehnoloogiatega.
3. Swarm on tihedalt seotud Ethereum'iga [21], mis võimaldab loomulikku integratsiooni teiste Web3 komponentidega tulevikus.

Swarm võrgule ligipääsu saamiseks on vajalik installeerida enda masinasse Bee sõlme [22] – need moodustavad Swarm võrgustikku.

On olemas sarnane süsteem - IPFS [23], mis on ehitatud samamoodi detsentraliseeritud arhitektuuri põhjal. IPFS ei sobinud rakenduse realiseerimiseks sellepärast, et IPFS võrgus andmevahetuse eest on vaja maksta, ning maksmine käib tavaliste raha ülekannetega, mis ei ole detsentraliseeritud. Transaktsioonid on hallatud tsentraliseeritud kujul. Seega detsentraliseeritud on ainult süsteemi arhitektuur ise ja andmevahetuse protsess, kuid süsteemi kui tervikut võib nimetada hübriidseks.

2.4 Tööprotsessi ülevaade

Töö valmis ligikaudu 9 kuu jooksul ajavahemikus august 2024 kuni aprill 2025. Selle aja jooksul tegeles autor aktiivselt lõputööga, ning sellele eelnes ka sissejuhatus meditsiiniinformaatika valdkonda. Ettevalmistustööga alustas autor 2024 aasta märtsis. Samal ajal lõputöö teema oli valitud koos lõputöö juhendajaga. Eeltöö sisaldas tervishoiuvaldkonnaga tutvumist tarkvaraarenduse tehnoloogiate vaatenurgast, selle teoreetilise baasi omandamist, teadustöödest varasemate ja tänapäeva trendide ja probleemide analüüsimist. Lisaks, sai autor tugeva kogemuse varem töötades meditsiinivaldkonna projektis põhitöökohal tarkvaraarenduse ettevõttes. Projekt oli tellitud PERH (Põhja-Eesti Regionaalhaigla) poolt.

Konsultatsioonid lõputöö juhendajaga toimusid regulaarselt läbi Teams-i vahenduse. Eeltöö jooksul toimunud konsultatsioonide käigus valideeris autor koos juhendajaga, et valdkonna uurimine edeneb õiges suunas ning jagas oma mõtteid ja esitas küsimusi. Põhitöö jooksul konsultatsioonid toimusid iganädalaselt. Konsultatsioonide ajal saadi tagasiside möödunud nädala tulemustest, lahendati tekkinud küsimusi ning arutati, mis on järgmine samm lõpptulemuse saavutamiseks. Lõputöö vältel oli autor pidevas suhtluses juhendaja ja kaasjuhendajaga, kõik osapooled olid tulemuste valmimisel alati kaasatud protsessi ning järgnevate sammude otsustamisse.

3. Tulemused

Peatükis 3.1 vaadeldakse süsteemi kasutajaid. Peatükis 3.2 kavandatakse süsteemi kasutusjuhtusid. Peatükis 3.3 selgitatakse, kuidas töötab terviseandmete versioneerimine ja peamise võtme kasutamine andmete kättesaamiseks. Peatükis 3.4 vaadeldakse isiklike terviseandmete hoidmise ja kasutamise loogikat ja põhiprintsiipe. Peatükis 3.5 esitletakse DICOM piltide laadimise, hoidmise ja kuvamise funktsionaalsust. Peatükis 3.6 kavandatakse FHIR ressursside kategoriseerimist. Peatükkides 3.7, 3.8 ja 3.9 keskendutakse süsteemi tehnilise ülesehituse selgitamise, arhitektuuri ülesehitust ja põhikomponente tutvustamise ning rakenduse komponendi struktuuri esitlemisele. Peatükis 3.10 tõlgendatakse terviseandmete struktuuri käsitlemist. Peatükis 3.11 esitletakse süsteemi testimise teostatavust.

3.1 Süsteemi kasutajad

Rakendusel äriloogika kohaselt eeldatakse kaks tüüpi põhikasutajaid – terviseandmete omanik ja tervishoiuteenuse osutaja. Rakenduse tehnilise poole pealt tähendab see kaks kasutajagruppe erinevate õigustega.

Tabel 2. Rakenduse kasutajad ja õigused

Kasutaja	**Õigused**
Terviseandmete omanik	(1) Enda terviseandmete vaatamine (2) Enda kohta andmete lisamine *Observation* FHIR ressurssi tüüpi *social-history* ja *activity* koodiga (3) Enda pereliikme ja/või eestkostetava andmetele ligipääsu lisamine (4) Enda pereliikme ja/või eestkostetava andmete vaatamine (5) Enda pereliikme ja/või eestkostetava kohta andmete lisamine *Observation FHIR* ressurssi tüüpi *social-history* ja *activity* koodiga
Tervishoiuteenuse osutaja	(1) Patsiendi andmete vaatamine juhul, kui patsient on jaganud ligipääsu enda andmetele (2) Patsiendi kohta andmete lisamine (3) Patsiendi uuendatud ligipääsu võtme (peamine võti) jagamine patsiendiga

Käesoleva töö raames autor keskendub isiklike terviseandmete omaniku tüübi kasutaja funktsionaalsusel ning temaga seotud võimalikke kasutusjuhtude kirjeldamisel isiklike terviseandmete omaniku kasutaja punktidele (1) ja (2).

Tüüpiline kasutaja ehk Persoona

Allpool kirjeldatakse rakenduse potentsiaalsete kasutajate tüübid, kasutajate motivatsioon ja põhjused rakenduse kasutamiseks. Sellest kirjeldusest kujunevad rakenduse persoonad [24]. Persoona aitab täpsemini identifitseerida rakenduse kasutajagruppe, kasutajate soovid ja eesmärgid.

Tabel 3. Rakenduse persoona Kadi

Nimi, sugu, amet, vanus, asukoht, iseloomustav kirjeldus	Kadi, naine, koduperenaine, vanus 37a, elab Tallinnas. Abielus, 2 last, enda vanaema hooldaja ja eestkostja. Üks lastest on nõrgema tervisega ning tihti satub arsti juurde, vajab erinevate ravimite tarvitamist.
Motivatsioon	Iga pereliikme tervise ülevaade on täiuslik, mis võimaldab paremini ja õigeaegselt jälgida ja märkida terviseseisundi muutusi, saada arsti visiidist maksimaalselt kasu tänu detailselt kogutud andmetele.
Ootused, vajadused, lootused	■ Nii enda, kui ka pereliikmete ja laste terviseandmed on terviklikud ning kättesaadavad ühest kohast, ilma erinevatest tervishoiuasutustest andmed küsimata; ■ Terviseseisundi muutusi on mugavam jälgida ja arsti visiidid on palju kasulikumad.
Mured, probleemid, hirmud	■ Isiklike terviseandmete osaliselt paiknemine erinevates tervishoiuasutustes, järelikult terviseinfo võib vahepeal kaotsi minna, terviseseisundi pilt ei ole terviklik; ■ Terviseseisundi detailse ülevaate puudumine; ■ Mõne tervishoiuasutuse süsteemi rikke tõttu võivad kogutud terviseandmed kaduma minna.

Persoona Kadi jaoks on väga oluline tema pere terviseseisund. Lisaks enda lastele, Kadi hoolitseb ka eaka vanaema eest. Seega eriti antud olukorras Kadi jaoks on tähtis, et tema ise saaks mugavalt ligipääsu vanaema terviseandmetele, saaks vanaema andmed arstiga jagada ning jälgida terviseseisundi muutusi. Sama eesmärk on Kadil ka laste puhul – nende terviseseisudi detailne ülevaade ja terviseandmete kättesaadavus.

Tabel 4. Rakenduse persoona Kaspar

Nimi, sugu, amet, vanus, asukoht, iseloomustav kirjeldus	Kaspar, mees, tarkvaraarendaja, vanus 28a, elab Tartus. Suhtes, lapsi ei ole. Lisaks inimlikele põhjustele olla terve jmt, Kaspar on teadlik andmete ja andmete privaatsuse tagamise tähtsusest.
Motivatsioon	Olla terve ja elada tervislikumalt ja pikemalt, mis võibki olla tagatud korrektse tervise eest hoolitsemisega ja tervise probleemide õigeaegse tuvastamisega.
Ootused, vajadused, lootused	■ Enda tervise eest hoolitsemine; ■ Elustiili ja harjumuste parendamine; ■ Tervikliku tervise ajaloo koostamine.
Mured, probleemid, hirmud	■ Piisavate terviseandmete puudumine elujooksul (eriti vanuses 18 – 35a); ■ Andmete puudumise tõttu terviseseisundi tervikliku pildi loomine ei õnnestu; ■ Järelikult, potentsiaalsed terviseprobleemid tulevikus võivad olla tuvastatud hiljem, kui oleks vaja.

Persoona Kaspar jaoks on oluline hoolitseda enda tervise eest ning õigeaegselt tuvastada potentsiaalseid probleeme. Kaspar plaanib luua oma pere ja kasvatada lapsi, ning selleks ta tahab olla terve, jälgida oma tervist regulaarselt.

3.2 Süsteemi kasutusjuhud

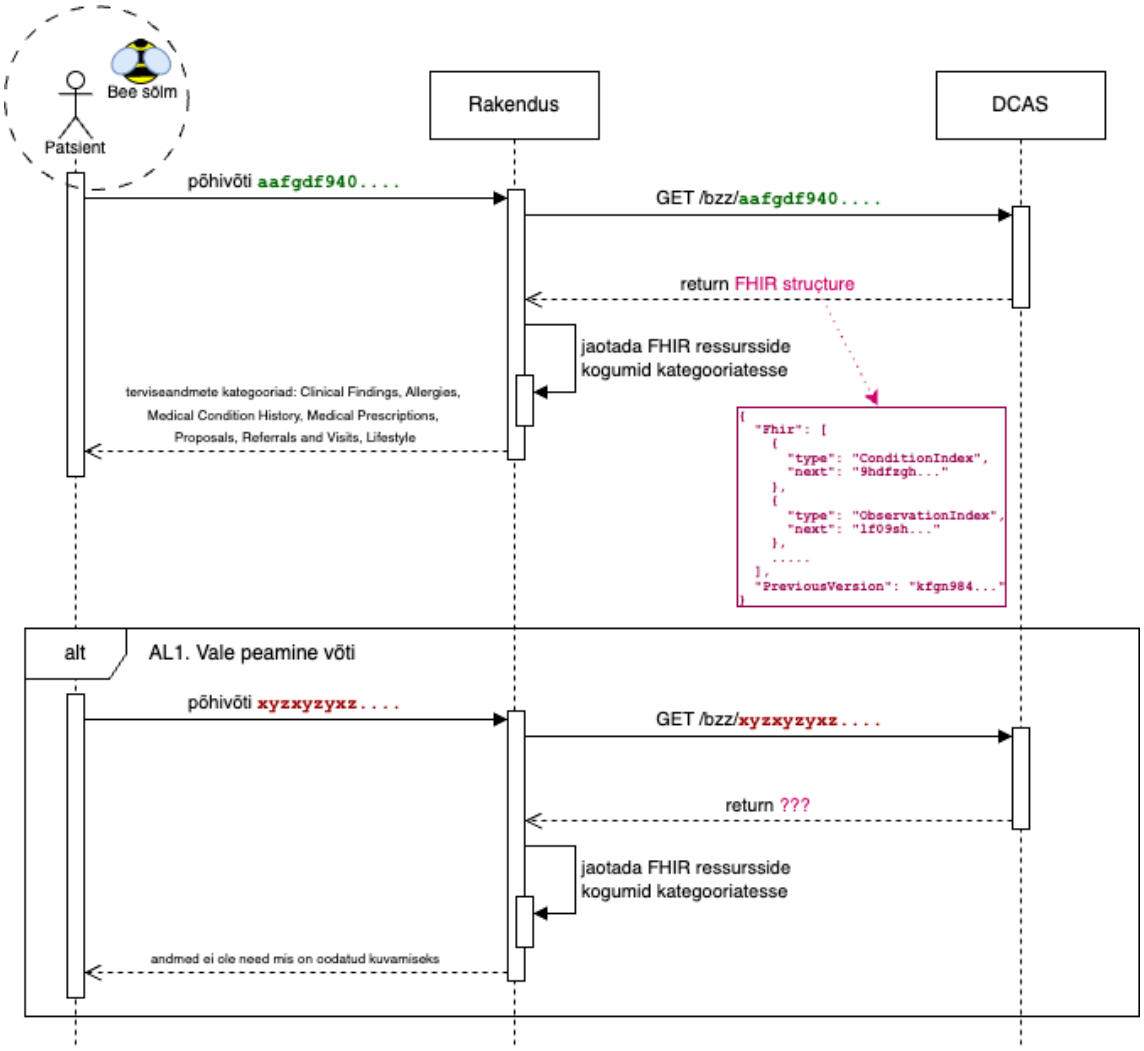
Käesolevas peatükis kirjeldab autor peamised kasutusjuhud, mida on võimalik läbida arendatud rakenduse praeguse etapi võimekuse raames.

KJ01 – Rakenduse pealehe avamine

Patsient avab enda seadmes rakenduse ning näeb rakenduse pealehe terviseandmete kategooriatega (Joonis 2).

- Tegutseja(d)
  - Patsient X
- Eeltingimused
  - Tegutseja seadmes peab olema installeeritud Bee sõlm
  - Tegutseja on Patsient
  - Tegutsejal peab olema peamine võti tema andmetele ligipääsemiseks
  - Andmed on selle peamise võtmega süsteemis salvestatud

- Järeltingimused
  - Tegutseja näeb rakenduse pealehe terviseandmete kategooriatega
  - Süsteem on saanud FHIR ressursside kogumid DCAS võrgust
  - Süsteem on jaotanud FHIR ressursside kogumid kategooriatesse
- Protsessi põhivoog
  - Tegutseja avab rakenduse
  - Tegutseja näeb terviseandmete kategooriad: Clinical Findings, Allergies, Medical Condition History, Medical Prescriptions, Proposals, Referrals, Visits, Lifestyle
  - Süsteem loeb tegutseja seadmes hoitud peamise võtme väärtust
  - Süsteem saadab andmete päringu DCAS süsteemi peamise võtme väärtuse järgi
  - Süsteem saab nimekirja FHIR ressursside kogumitest
  - Süsteem jaotab FHIR ressursside kogumid kategooriatesse
- Alternatiivsed protsessivood
  1. Alternatiiv 1. Vale peamine võti
     - AL1_1. Süsteem saadab andmete päringu DCAS süsteemi etteantud võtme järgi
     - AL1_2. Süsteem kuvab tegutsejale vastusena saadud tulemust
     - AL1_3. Tegutseja ei näe oodatud andmed



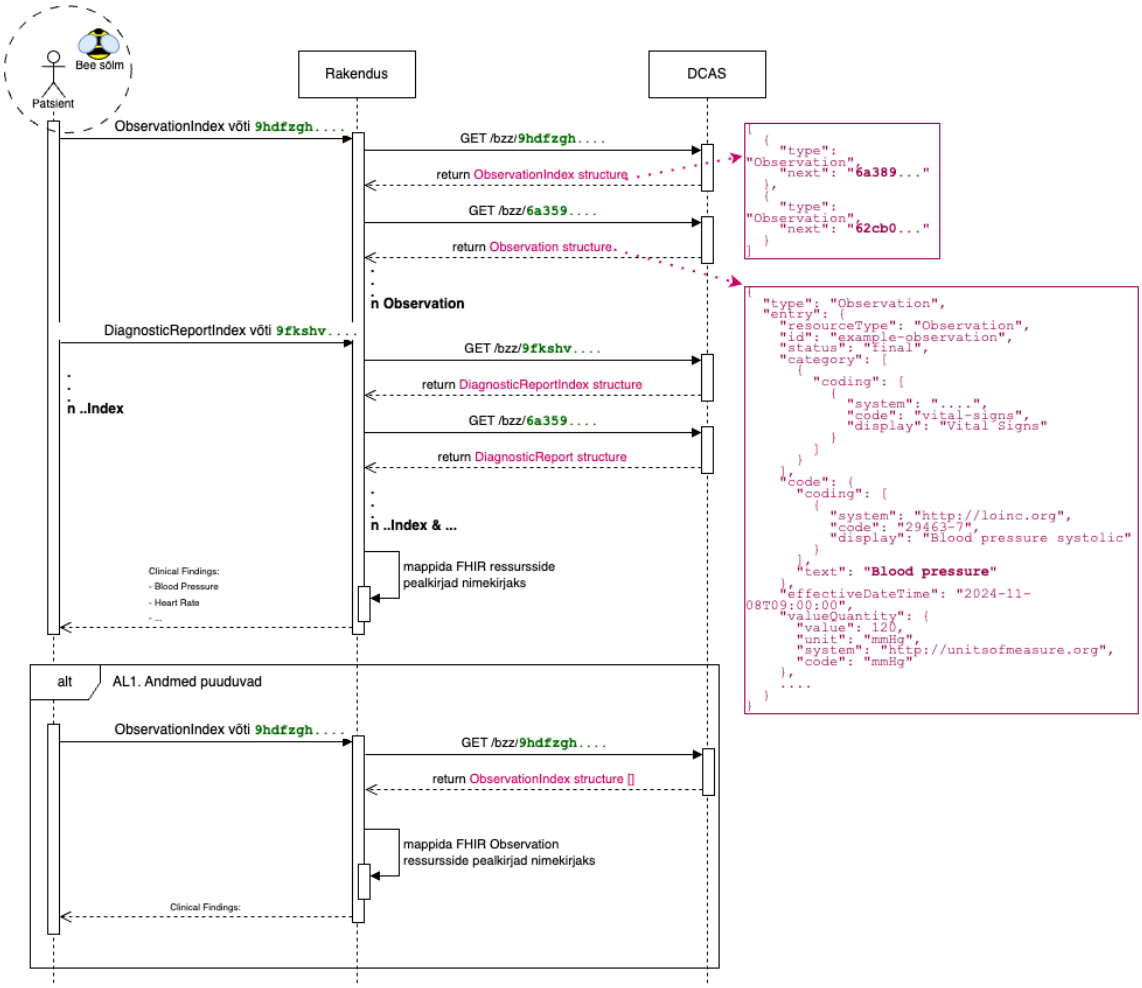
Joonis 2. *Kasutusjuhu KJ01 jadadiagramm*

KJ02 – Terviseandmete kategooria avamine

Patsient avab valitud kategooria ning näeb kategooria alla kuuluvad terviseandmete (FHIR ressursside) kogumid (Joonis 3).

- Tegutseja(d)
  - Patsient X
- Eeltingimused
  - Tegutseja asub rakenduse pealehel
  - Tegutseja näeb terviseandmed kategooriad: Clinical Findings, Allergies, Medical Condition History, Medical Prescriptions, Proposals, Referrals, Visits, Lifestyle
- Järeltingimused
  - Tegutseja näeb pealehel valitud kategooria FHIR ressursside kogumid

- Protsessi põhivoog
  - Tegutseja valib terviseandmete kategooria A
  - Süsteem kuvab selle kategooria alla kuuluvad FHIR ressursside kogumid
- Alternatiivsed protsessivood
  1. Alternatiiv 1. Andmed kategooria all puuduvad
     - AL1_1. Süsteem ei kuva midagi valitud kategooria all
     - AL1_2. Tegutseja ei näe ühtegi FHIR ressursside kogumit



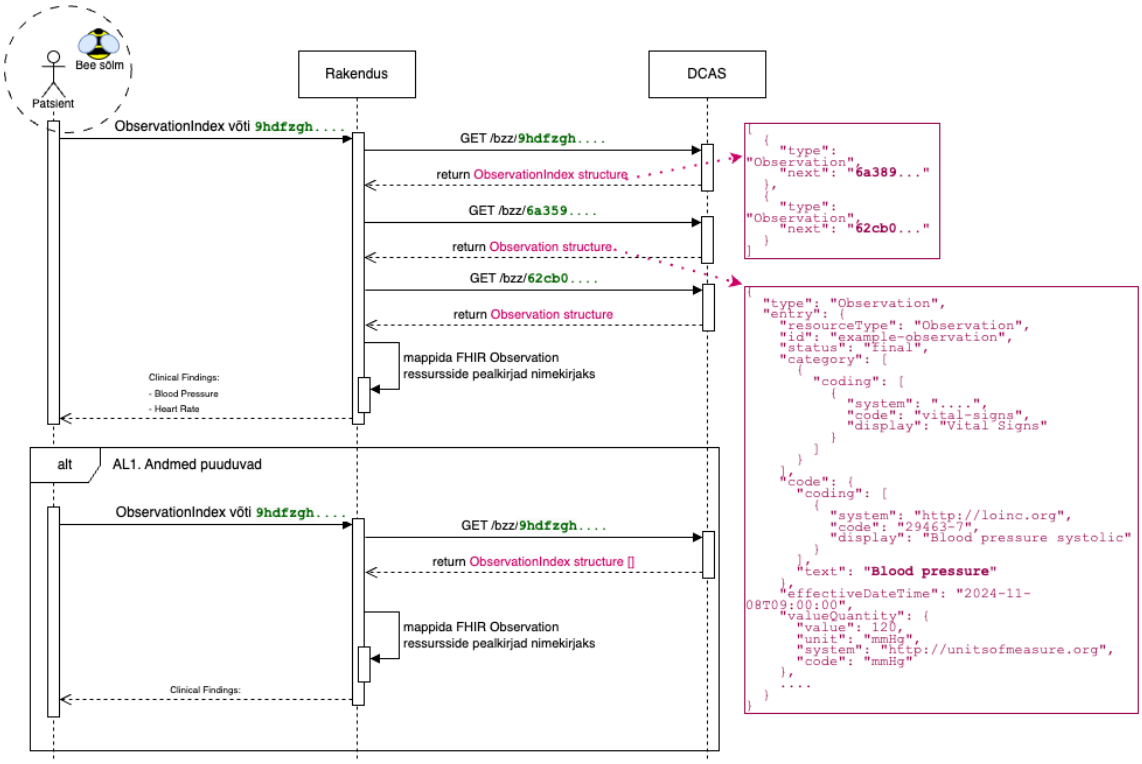
Joonis 3. *Kasutusjuhu KJ02 jadadiagramm*

KJ03 - FHIR ressursside kogumi avamine

Patsient avab valitud terviseandmete kogumi ning näeb sinna kuuluvad eraldiseisvad ühikud (üksikud FHIR ressurssid) (Joonis 4).

- Tegutseja(d)
  - Patsient X

- Eeltingimused
  - Tegutseja on valinud ühte terviseandmete kategooriatest: Clinical Findings, Allergies, Medical Condition History, Medical Prescriptions, Proposals, Referrals, Visits või Lifestyle
  - Tegutseja näeb pealehel valitud kategooria FHIR ressursside kogumid
- Järeltingimused
  - Tegutseja näeb valitud FHIR ressursside kogumi alla kuuluvad üksikud FHIR ressurssid
  - Üksikud FHIR ressurssid on klikkitavad detailsema info kuvamiseks
- Protsessi põhivoog
  - Tegutseja valib terviseandmete kogumi A
  - Süsteem kuvab selle kogumi kuuluvad FHIR ressursside ühikud
- Alternatiivsed protsessivood
  1. Alternatiiv 1. Andmed kogumi sees puuduvad
     - AL1_1. Süsteem ei kuva midagi valitud kogumi sees
     - AL1_2. Tegutseja ei näe ühtegi FHIR ressurssi ühikut



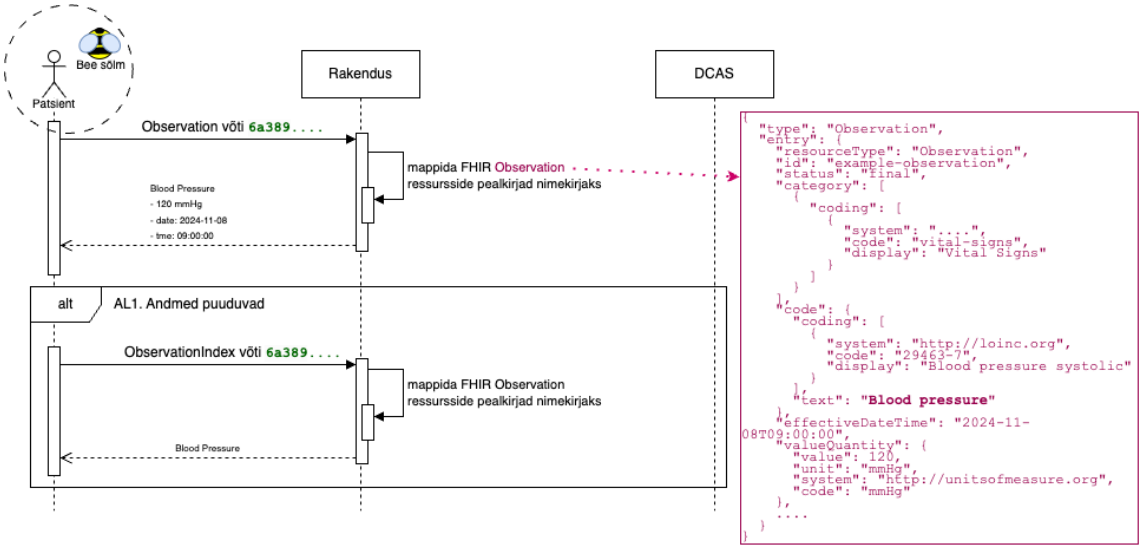
Joonis 4. *Kasutusjuhu KJ03 jadadiagramm*

KJ04 – FHIR ressursside kogumi ühiku avamine

Patsient avab valitud terviseandmete FHIR ressurssi ühiku ning näeb detailset kirjeldust

(Joonis 5).

- Tegutseja(d)
  - Patsient X
- Eeltingimused
  - Tegutseja näeb valitud FHIR ressursside kogumi alla kuuluvad üksikud FHIR ressurssid
  - Üksikud FHIR ressurssid on klikkitavad detailsema info kuvamiseks
- Järeltingimused
  - Tegutseja näeb valitud FHIR ressurssi ühiku detailse info
- Protsessi põhivoog
  - Tegutseja vajutab soovitud FHIR ressurssi ühikule
  - Süsteem kuvab valitud FHIR ressurssi ühikus sisalduva info
- Alternatiivsed protsessivood
  1. Alternatiiv 1. Info ühiku sees puudub
     - AL1_1. Süsteem ei kuva detailset infot valitud FHIR ressurssi ühikus
     - AL1_2. Tegutseja ei näe infot valitud FHIR ressurssi ühiku sees

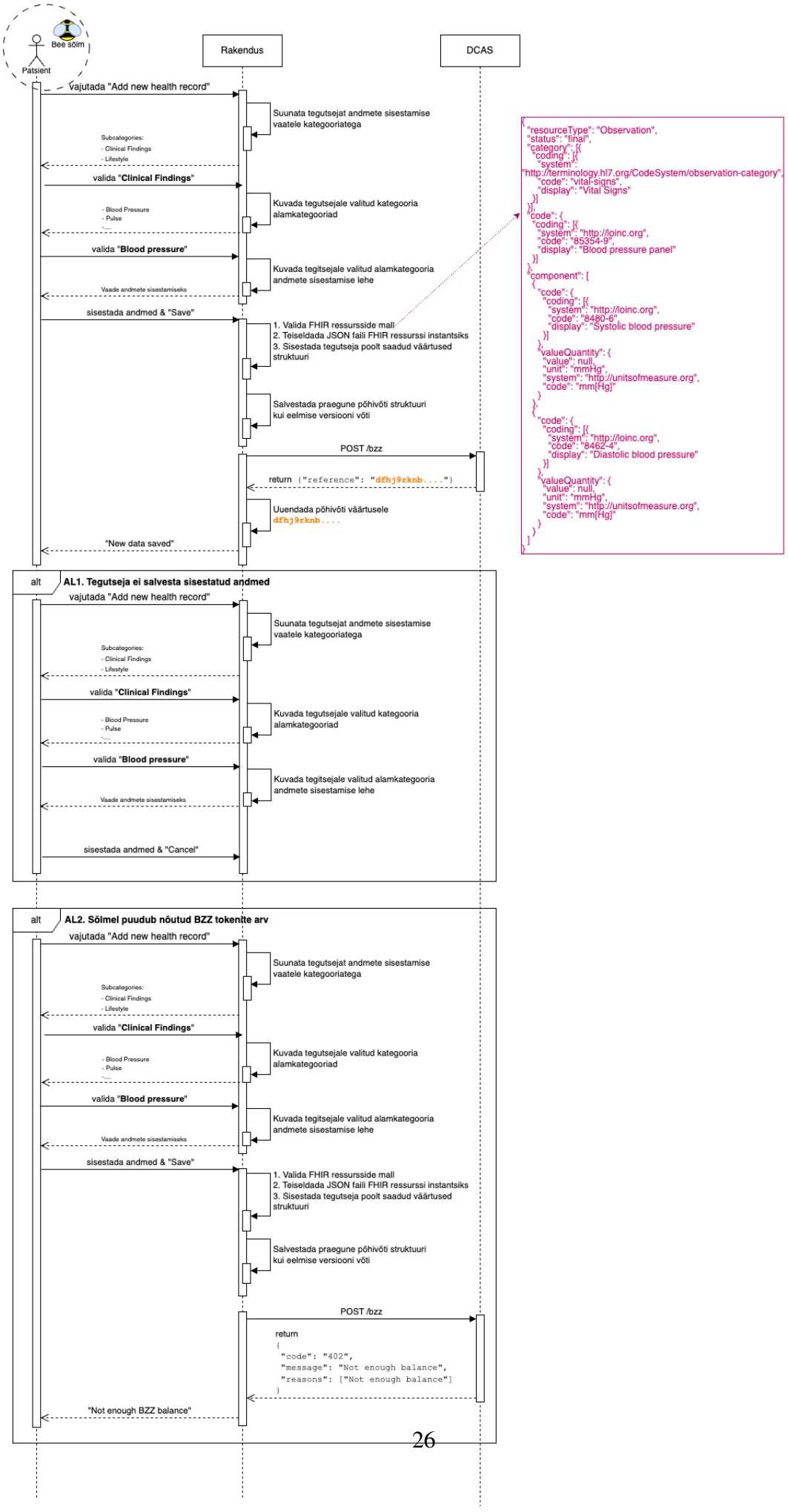


Joonis 5. *Kasutusjuhu KJ04 jadadiagramm*

KJ05 – Terviseandmete sisestamine Patsiendi rollis

Antud kasutusjuht keskendub konkreetselt elustiili ja -harjumuste andmete sisestamisele, kuna ainult need andmed saab patsient iseseisvalt enda kohta sisestada. Andmete sisestamise tulemusena näeb patsient lisandunud terviseandmete ühiku Elustiili kategooria all. Ligipääsu peamist võtit uuendatakse ja salvestatakse seadmesse (Joonis 6).

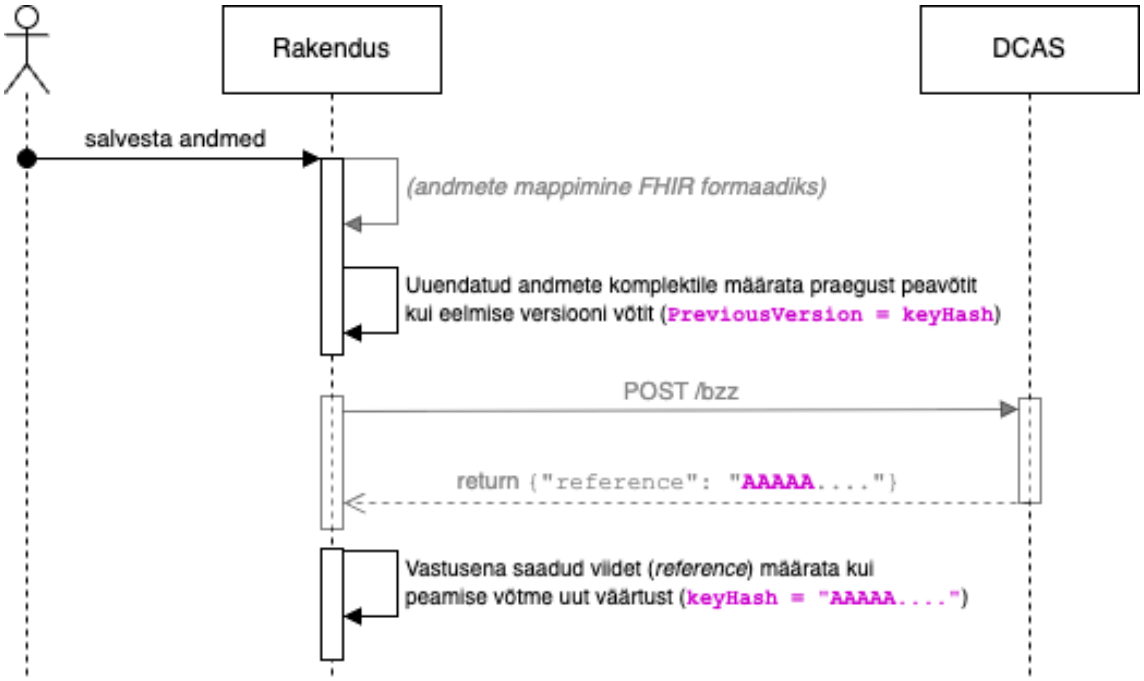
- Tegutseja(d)
  - Patsient X
- Eeltingimused
  - Tegutseja asub rakenduse pealehel
  - Bee sõlmel on kehtiv postmark ehk *Postage Stamp*
- Järeltingimused
  - Ligipääsu peamine võti on uuendatud ja salvestatud seadmesse
  - Sisestatud andmed on nähtavad Elustiili (Lifestyle) kategooria all
- Protsessi põhivoog
  - Tegutseja vajutab „Add new health record“ nuppu
  - Süsteem suunab tegutsejat andmete sisestamise vaatele
  - Tegutseja valib, millise kategooria andmed sisestada
  - Tegutseja trükib tekstivälja sisse nõutud andmed
  - Tegutseja vajutab „Save“ nuppu
  - Süsteem valib FHIR ressurssi malli JSON kujul, mis vastab sisestatud andmete tüübile
  - Süsteem moodustab andmetest JSON kujul FHIR ressurssi objekti
  - Süsteem sisestab struktuuri tegutseja poolt saadud ressurssi väärtused
  - Süsteem salvestab viide andmete eelmisele (ehk praegusele, enne salvestamist) versioonile
  - Süsteem salvestab andmete uue versiooni DCAS süsteemi
  - Süsteem saab salvestamise päringu vastusena uuendatud andmete ligipääsu peamise võtme
  - Süsteem salvestab uue peamise võtme seadmesse
  - Süsteem kuvab informatiivse sõnumi, et andmed on salvestatud
- Alternatiivsed protsessivood
  1. Alternatiiv 1. Tegutseja ei salvesta sisestatud andmed
     - AL1_1. Tegutseja vajutab „Cancel“ nuppu „Save“ nuppu asemel
     - AL1_2. Süsteem paneb andmete sisestamise vaade kinni
  2. Alternatiiv 2. Sõlmel puudub kehtiv postmark
     - AL2_1. Tegutseja vajutab „Save“ nuppu
     - AL2_2. Süsteem saadab andmete salvestamise päringu DCAS süsteemi
     - AL2_3. Süsteem saab päringu vastusena vea sõnumiga, et puudub postmark andmete salvestamiseks
     - AL2_4. Süsteem kuvab vastava veasõnumi kasutajale


Bee sõlm
Patsient
Rakendus
DCAS
vajutada "Add new health record"
Suunata tegutsejat andmete sisestamise vaatele kategooriatega
Subcategories:
- Clinical Findings
- Lifestyle
valida "Clinical Findings"
Kuvada tegutsejale valitud kategooria alamkategooriad
- Blood Pressure
- Pulse
valida "Blood pressure"
Kuvada tegitsejale valitud alamkategooria andmete sisestamise lehe
Vaade andmete sisestamiseks
sisestada andmed & "Save"
1. Valida FHIR ressursside mall
2. Teiseldada JSON faili FHIR ressurssi instantsiks
3. Sisestada tegutseja poolt saadud väärtused struktuuri
Salvestada praegune põhivõti struktuuri kui eelmise versiooni võti
POST /bzz
return {"reference": "dfhj9rknb...."}
Uuendada põhivõti väärtusele dfhj9rknb....
"New data saved"
{
"resourceType": "Observation",
"status": "final",
"category": [{
"coding": [{
"system": "http://terminology.hl7.org/CodeSystem/observation-category",
"code": "vital-signs",
"display": "Vital Signs"
}]
}],
"code": {
"coding": [{
"system": "http://loinc.org",
"code": "85354-9",
"display": "Blood pressure panel"
}]
},
"component": [
{
"code": {
"coding": [{
"system": "http://loinc.org",
"code": "8480-6",
"display": "Systolic blood pressure"
}]
},
"valueQuantity": {
"value": null,
"unit": "mmHg",
"system": "http://unitsofmeasure.org",
"code": "mm[Hg]"
}
},
{
"code": {
"coding": [{
"system": "http://loinc.org",
"code": "8462-4",
"display": "Diastolic blood pressure"
}]
},
"valueQuantity": {
"value": null,
"unit": "mmHg",
"system": "http://unitsofmeasure.org",
"code": "mm[Hg]"
}
}
]
}
alt
AL1. Tegutseja ei salvesta sisestatud andmed
vajutada "Add new health record"
Suunata tegutsejat andmete sisestamise vaatele kategooriatega
Subcategories:
- Clinical Findings
- Lifestyle
valida "Clinical Findings"
Kuvada tegutsejale valitud kategooria alamkategooriad
- Blood Pressure
- Pulse
valida "Blood pressure"
Kuvada tegitsejale valitud alamkategooria andmete sisestamise lehe
Vaade andmete sisestamiseks
sisestada andmed & "Cancel"
alt
AL2. Sõlmel puudub nõutud BZZ tokenite arv
vajutada "Add new health record"
Suunata tegutsejat andmete sisestamise vaatele kategooriatega
Subcategories:
- Clinical Findings
- Lifestyle
valida "Clinical Findings"
Kuvada tegutsejale valitud kategooria alamkategooriad
- Blood Pressure
- Pulse
valida "Blood pressure"
Kuvada tegitsejale valitud alamkategooria andmete sisestamise lehe
Vaade andmete sisestamiseks
sisestada andmed & "Save"
1. Valida FHIR ressursside mall
2. Teiseldada JSON faili FHIR ressurssi instantsiks
3. Sisestada tegutseja poolt saadud väärtused struktuuri
Salvestada praegune põhivõti struktuuri kui eelmise versiooni võti
POST /bzz
return
{
"code": "402",
"message": "Not enough balance",
"reasons": ["Not enough balance"]
}
"Not enough BZZ balance"

3.3 Peamine võti ja terviseandmete versioneerimine

Detsentraliseeritud sisu-adresseeritavas salvestusvõrgus andmete komplekti muudatusi käsitletakse kui täiesti uusi andmeversioone. Käesoleva töö raames valmis tehtud rakenduses sai arendatud andmete komplekti versioneerimine.

Andmete komplekti versioneerimise loogika seisneb selles, et iga andmete komplekti muudatusega saab rakendus DCAS süsteemist vastusena uue ligipääsu võtme (*reference*). Peamise võtme väärtus uuendatakse saadud uue ligipääsu võtme väärtuse vastu. Andmete komplekti külge lisatakse viide eelmisele andmete komplekti versioonile. Joonisel 7 näidatakse, kuidas ja millises loogilises järjekorras toimub andmete komplekti eelmise versiooni määramine ning uuendatud peamise võtme väärtuse kättesaamine.



Joonis 7. *Andmete komplekti uuendamine ja peamise võtme väärtuse uuendamine*

Single Owner Chunks

Single Owner Chunk ehk ühe omaniku andmetükk on Swarm detsentraliseeritud sisu-adresseeritavas salvestusvõrgus spetsiaalne andmeühik, mille saab salvestada ja millele pääseb ligi ainult selle omanik. Omanik (*owner*) on SOC looja ning SOC loomisel krüpteerib andmetükki oma Ethereum'i privaatvõtmega. [25].

Viktor Trón oma raamatus "The Book of Swarm" detailselt kirjeldab Single Owner Chunk loogika põhimõtteid ja tehnilise teostatavuse detaile [26].

Lõputöö raames autor katsetas *Single Owner Chunk* implementeerimise võimaluse oma rakenduses. Implementeerimine on arendatud eraldi Git haru peal [27].

3.4 Isiklike terviseandmete hoidmine ja kasutamine

Käesoleva töö raames tehtud rakendus võimaldab salvestada andmed süsteemi JSON formaadis vastavalt FHIR standardile. Selleks kasutatakse Bee API meetod `POST /bzz` [28]. Andmete vaatamiseks kasutatakse Bee API meetod `GET /bzz/reference` [29], mis tagastab andmete struktuuri, mis oli varem salvestatud Swarm võrgustikku.

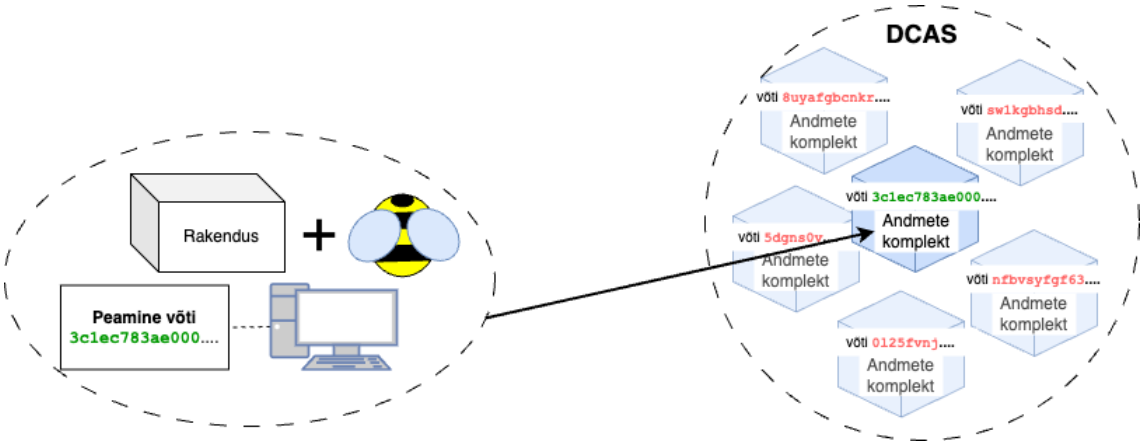
Joonis 8. *POST /bzz ja GET /bzz/{reference} päringud Bee API's*

Joonisel 8 diagrammil näidatakse, kuidas vastavate `POST` ja `GET` päringute saatmine Swarm võrgustikku toimub Bee API kaudu. Detailsemad kasutusjuhud konkreetsete andmete komplektidega on kirjeldatud alapeatükis 3.2.

JSON formaat oli valitud antud töö raames mugavuse eesmärgil selleks, et valideerida andmete sisu korrektsust ka ilma vahekihilist rakendust kasutamata. Swarm sõlmede funktsionaalsus pakub ka andmete tükkide [30] ja andmete baite [31] üleslaadimist. Tükeldatud ja/või baitide kujul lähenemine vähendab andmete mahtu ning järelikult vähendab ka koormust kogu süsteemile. Andmed on hoitud ajaloona ning andmete komplekt uuendatakse tervikuna, kui toimub uute andmete lisamine. Peamine võti andmete komplektile viidab komplekti viimasele versioonile, ning uuendatakse ja salvestatakse uuesti koos andmete komplekti uuendamisega. Kehtival andmete komplekti versioonil on olemas ka viide eelmisele versioonile.

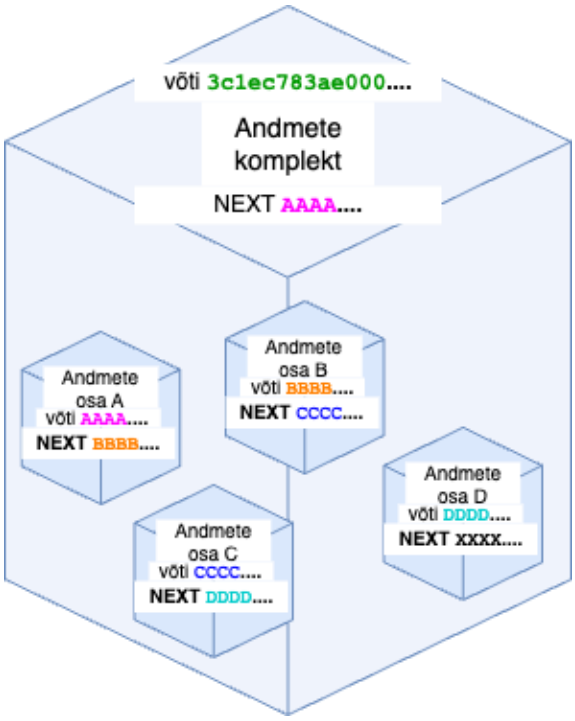
Andmetele ligipääsu mehhanism ja turvalisus

Andmetele ligipääsemiseks on vaja unikaalset võtit (peamine võti, põhivõti), mis vastab konkreetsetele andmete segmentidele. Terviseandmete omaniku puhul on peamine võti seotud tema enda andmete komplektiga. Tulevastes arenduse etappides peamine võti on hoitud terviseandmete omaniku isiklikus seadmes riistvara tasemel krüpteeritud kujul. Praeguses arenduse etapis peamise võtme väärtus salvestatakse rakenduses. Peamine võti võimaldab juurdepääsu andmete struktuurile, kuid mitte kõikidele andmetele korraga (Joonis 9).



Joonis 9. *Andmete komplekti ligipääsemine peamise võtmega*

Andmete komplekti sees võivad olla täiendavad alamosad (nt haigusloo erinevad osad, röntgenipildid, retseptid), millel on eraldi võtmed. Nendele alamosadele viitavad teistes andmete tükkides hoitud võtmed. Need struktuuri sisemised võtmed võimaldavad juurdepääsu ainult määratud andmete komplekti alamosale, kuid mitte tervele terviseandmete omaniku andmete komplektile (Joonis 10).



Joonis 10. *Andmete komplekti alamosad võtmetega*

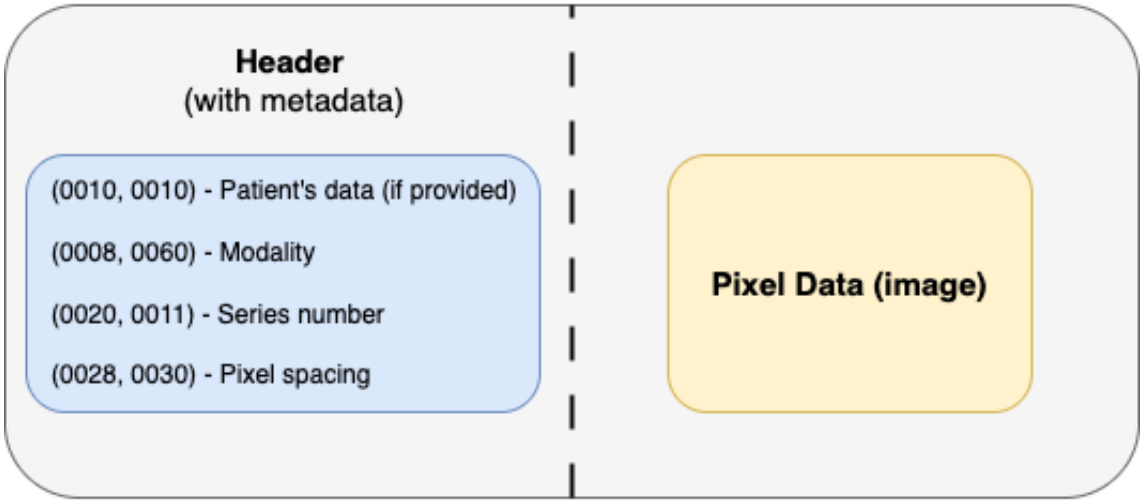
3.5 DICOM pildiandmed

DICOM pildiandmed on rahvusvaheline standard, mis oli välja töötatud selleks, et võimaldada meditsiiniliste kujutiste ja nendega seotud metaandmete ühtset salvestamist, edastamist ja töötlemist [8]. See standard määratleb nii faili struktuuri kui ka suhtlusprotokolli erinevate meditsiiniliste süsteemide vahel, nagu näiteks pildistamisseadmed (nt MRI, CT, röntgeniaparaadid), haiglainfosüsteemid ja -tööjaamad.

DICOM failid ei koosne ainult pildiandmetest, vaid faili koosseisu kuuluvad ka metaandmed, mis sisaldavad tehtud uuringuga seotud informatsiooni. Näiteks, uuringu üksikasjad (kuupäev, seade, parameetrid), anatoomiline piirkond, viited varasematele uuringutele ja muu kliiniliselt oluline info. Selline terviklik andmete komplekt võimaldab tervishoiuteenuse osutajatel ja meditsiini valdkonna tarkvarasüsteemidel andmeid automaatselt töödelda standardiseeritud kujul [32].

DICOM formaadis kasutatakse eristruktuuri, mis koosneb kahest osast: päise ja pildi sisu. Päise, (*header*) sisaldab metaandmeid. Pildi sisu (*pixel data*) sisaldab pildi pikslite andmed (vt Joonis 11). Iga andmeelement on kodeeritud nn *tag-value* paaridena (*Group, Element*). See muudab DICOM andmed masinloetavaks ja samas paindlikuks erinevate tootjate seadmete vahelisel suhtlusel [33] tänu standardiseeritud formaadile. Näiteks, võib DICOM päises olla kirje (0010,0010) – Patient's Name, mis sisaldab patsiendi nime informatsiooni,

või (0008,0060) – Modality, mis määrab kujutise tüübi (nt "MR" või "CT").



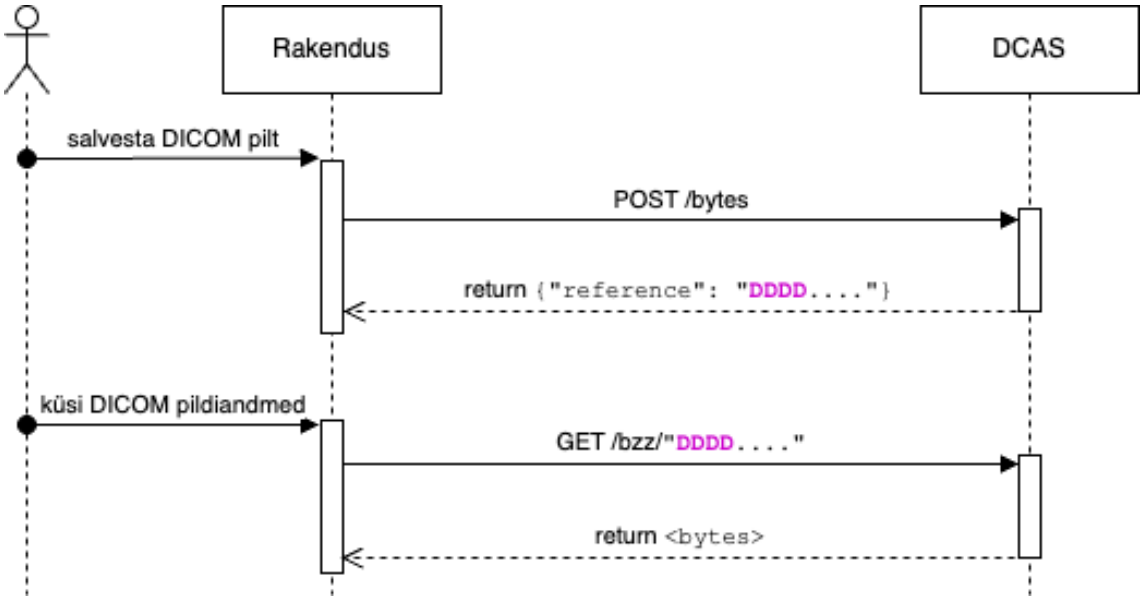
Joonis 11. *DICOM faili struktuur*

Kuna DICOM on spetsialiseeritud formaat, siis ei saa DICOM formaadis faile avada tavalise pilditöötlustarkvara abil nagu JPEG või PNG formaadi pildiandmete korral. DICOM pildiandmete faili avamiseks on vaja DICOM formaati toetavaid tööriistu. Selleks on näiteks online-tööriist DICOM Viewer [34], mis oli kasutatud käesoleva töö raames.

Lõputöö raames oli vajalik veenduda, et detsentraliseeritud sisu-adresseeritavast salvestusvõrgust tagasi saadud ja kokku pandud faili sisu jääb korrektseks ning ei ole andmete salvestamise ja tagasi pärimise käigus rikutud. Selleks oli kasutatud online-tööriist DICOM Viewer [34]. Tagasi saadud DICOM faili terviklikkuse ja korrektsuse valideerimiseks olid võrreldud omavahel esialgse DICOM pildiandmete faili sisu ja selle faili taastatud variant. DICOM faili taastatud variant oli saadud tagasi detsentraliseeritud sisu-adresseeritavast salvestusvõrgust. Võrdlemiseks on lisatud ekraanitõmmised esialgsest DICOM pildi vaatest (Lisa 3) ja selle taastatud variandist, mis oli avatud pärast detsentraliseeritud sisu-adresseeritavast salvestusvõrgust faili tagasi kokkupanemist (Lisa 4).

DICOM pildiandmete salvestamine

DICOM pildiandmete üleslaadimiseks on vaja meedia fail enne konverteerida baitide massiiviks ning salvestada detsentraliseeritud sisu-adresseeritavas salvestusvõrgus baitide kujul. Selleks kasutatakse Bee API meetod `POST /bytes` [31]. Pildiandmete kättesaamiseks baitide kujul kasutatakse Bee API meetod `GET /bytes/reference` [35]. Edaspidi baite on võimalik konverteerida tagasi *.dcm* laiendusega failiks.



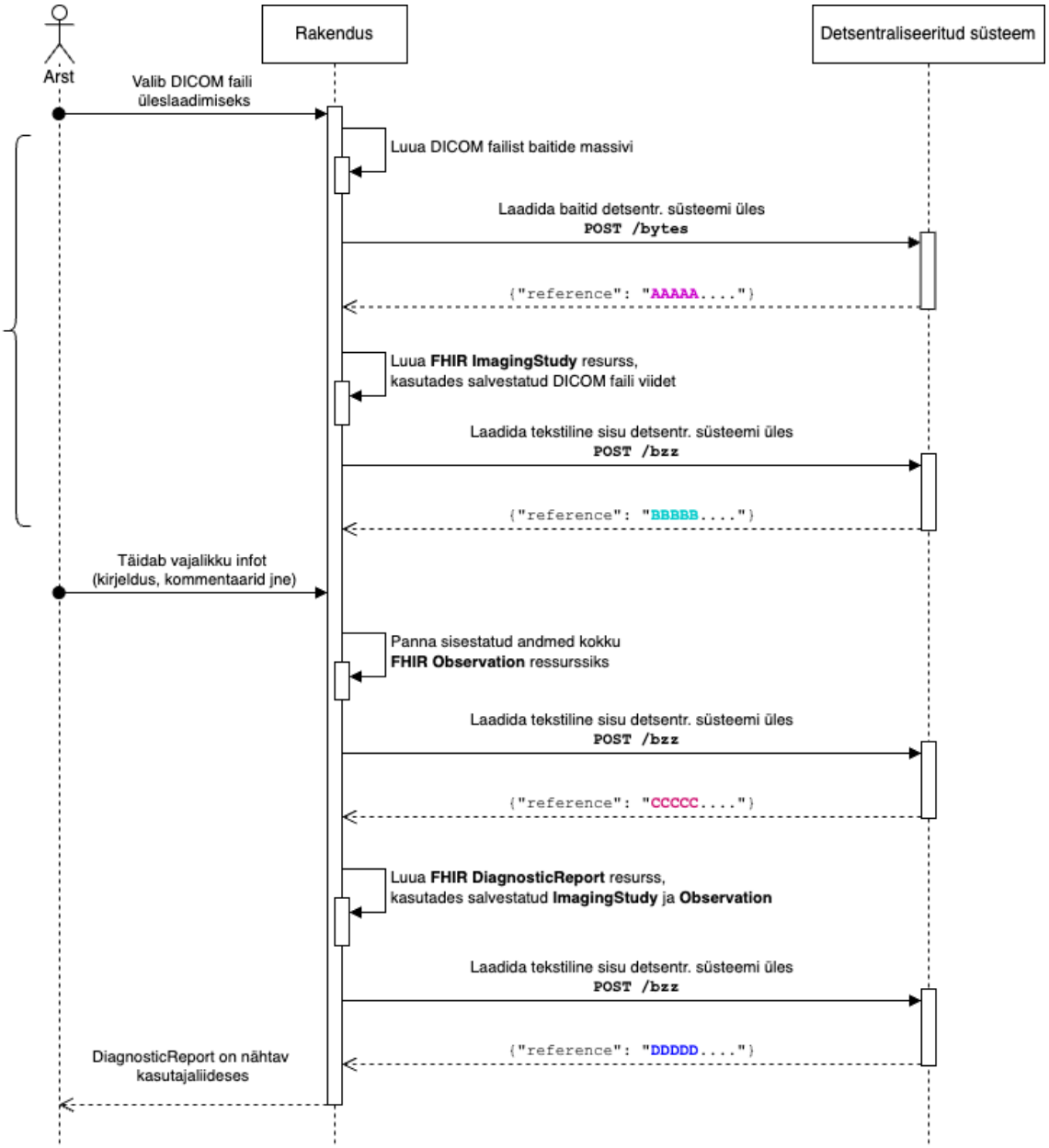
Joonis 12. *POST /bytes ja GET /bytes/{reference} päringud Bee API's*

Joonis 12 diagrammil näidatakse, kuidas vastavate `POST` ja `GET` päringute saatmine Swarm võrgustikku toimub Bee API kaudu. Detailsem DICOM pildiga seotud andmete komplekti loomise kirjeldus on allpool koos Joonis 13 diagrammiga.

DICOM pildi seos uuringu andmetega

DICOM pilt on FHIR ressursside struktuuris osa andmete komplektist, milles on omavahel seotud FHIR ressurssid:

- ImagingStudy - sisaldab informatsiooni (*meta-data*)ja viiteid DICOM piltidele;
- Observation - arsti poolne kirjeldus patsiendi seisundist, millele on lisatud DICOM pilt;
- DiagnosticReport - uuringu informatsiooni ja andmete kokku võtmiseks, viidab seotud ImagingStudy ja Observation ressurssidele.



Joonis 13. *Jadadiagramm uuringu ressurssi loomise protsessist koos DICOM faili üleslaadimisega*

Joonis 13 kirjeldab uuringu andmete komplekti loomise protsessi, mis sisaldab DICOM pildiandmeid. Andmete komplekti loomiseks on vaja:

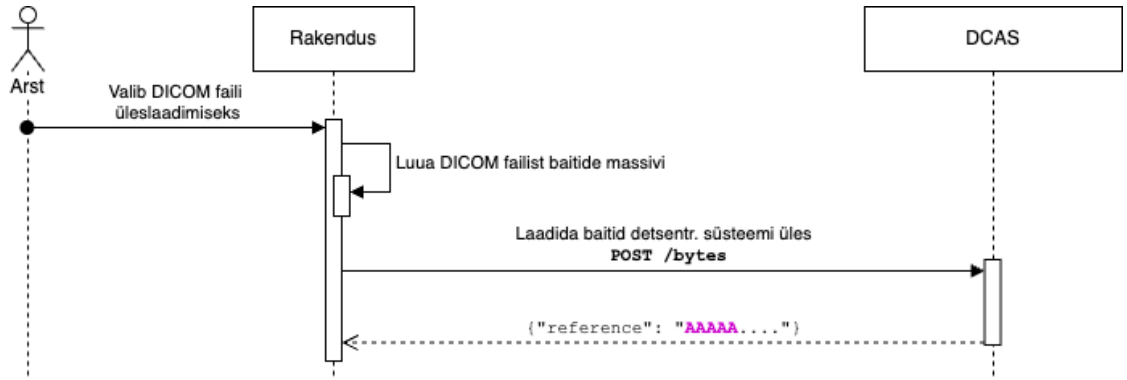
1. salvestada DICOM pilt detsentraliseeritud sisu-adresseeritavas salvestusvõrgus,
2. luua ja salvestada kolm FHIR ressurssi: ImaginStudy DICOM pildile viitamiseks, Observation arsti poolsete märkuste hoidmiseks, DiagnosticReport tehtud uuringu andmete kokku viitamiseks.

Uuringu andmete koos DICOM pildiandmetega loomise protsessi saab jagada neljaks osaks.

Iga sammu tulemusena andmed on salvestatud detsentraliseeritud sisu-adresseeritavas salvestusvõrgus. Salvestatud andmetest moodustatakse viide ehk võti andmetükile.

Samm 1 - DICOM faili salvestamine

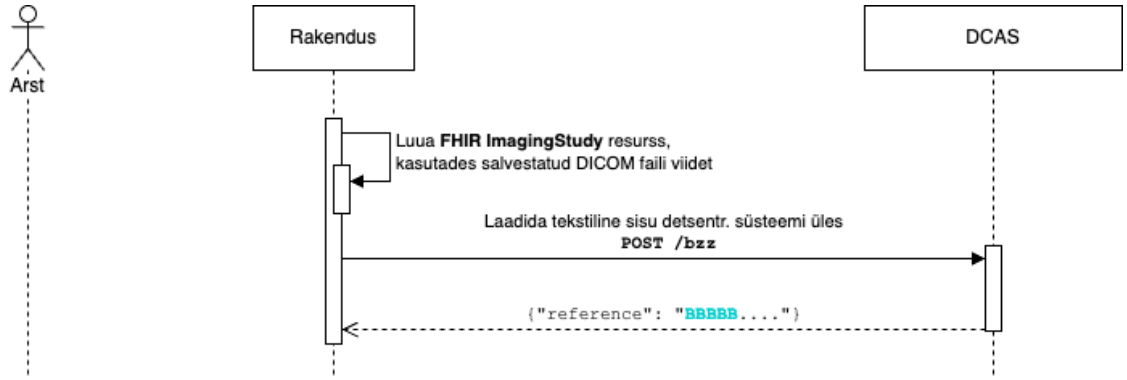
Kõigepealt, detsentraliseeritud sisu-adresseeritavas salvestusvõrgus salvestatakse DICOM fail, mis on seotud tehtud uuringuga. Joonisel 14 näidatakse, kuidas DICOM faili lisatakse süsteemi rakenduse kaudu, rakenduse sees failist luuakse baitide massiiv ning andmed baitide kujul saadetakse `POST /bytes` päringuga DCAS süsteemi.



Joonis 14. *DICOM faili salvestamine DCAS süsteemis baitide kujul*

Samm 2 - ImagingStudy FHIR ressurssi loomine

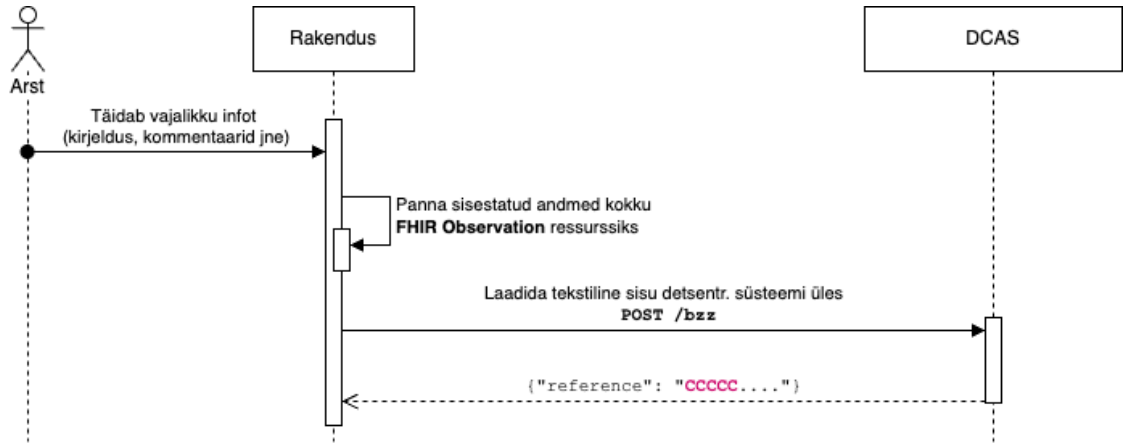
Järgmise sammuna luuakse programmaatiliselt ning salvestatakse detsentraliseeritud sisu-adresseeritavas salvestusvõrgus ImagingStudy FHIR ressurss, mille külge enne salvestamist on lisatud varem DCAS süsteemis salvestatud DICOM faili viide (vt Joonis 15). ImagingStudy FHIR ressurssi detailne sisu on kirjeldatud Lisas 5.



Joonis 15. *ImagingStudy FHIR ressurssi salvestamine DCAS süsteemis*

Samm 3 - Observation FHIR ressurssi loomine

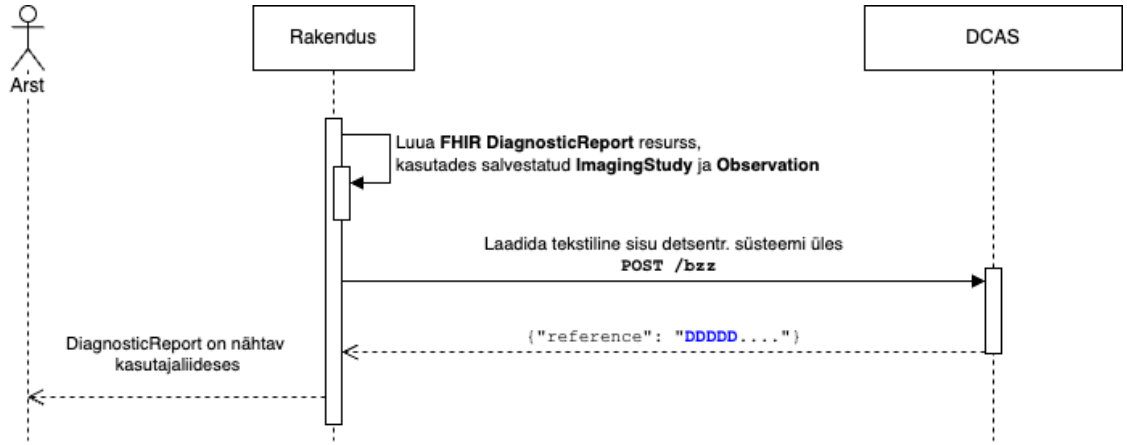
Järgmise sammuna luuakse programmaatiliselt ning salvestatakse detsentraliseeritud sisu-adresseeritavas salvestusvõrgus Observation FHIR ressurss, mille külge enne salvestamist on lisatud varem DCAS süsteemis salvestatud ImagingStudy ressurssi viide (vt Joonis 16). Observation FHIR ressurssi detailne sisu on kirjeldatud Lisas 6.



Joonis 16. *Observation FHIR ressurssi salvestamine DCAS süsteemis*

Samm 4 - DiagnosticReport FHIR ressurssi loomine

Viimase sammuna luuakse programmaatiliselt ning salvestatakse detsentraliseeritud sisu-adresseeritavas salvestusvõrgus DiagnosticReport FHIR ressurss, mille külge enne salvestamist on lisatud varem DCAS süsteemis salvestatud Observation ja ImagingStudy FHIR ressursside viited (vt Joonis 17). DiagnosticReport FHIR ressurssi detailne sisu on kirjeldatud Lisas 7.



Joonis 17. *DiagnosticReport FHIR ressurssi salvestamine DCAS süsteemis*

3.6 FHIR ressursside kategoriseerimine

Käesoleva töö üheks väljundiks on näidata, et pakutud süsteem võimaldab tagada kasutajatele tuttava keskkonda. Isiklikud terviseandmed on hoitud süsteemis FHIR standardi kujul, seetõttu FHIR ressursside tüüpe kategoriseerimiseks kasutas autor ressursside funktsionaalsete eesmärkide juhendit [36].

Tabel 5. FHIR ressursside kategoriseerimine

Kategooria (eng)	**Kategooria (est)**	**FHIR ressurss(id)**
Clinical Findings	Analüüside ja uuringute tulemused	Observation, DiagnosticReport, ImagingStudy
Allergies	Allergiad	AllergyIntolerance
Medical Condition History	Terviseseisundi ajalugu	Condition, FamilyMemberCondition
Medical Prescriptions	Retseptid	MedicationAdministration, MedicationState, MedicationRequest, Medication
Proposals	Soovitused	NutritionOrder, SupplyRequest
Referrals	Saatekirjad	ServiceRequest
Visits	Vastuvõtud	Encounter
Lifestyle	Elustiil	Observation

Allpool toob autor välja iga kategooria kasutusjuhu detailsema kirjelduse koos Andmevaaturi [37] ekvivalentse vaate näidetega. Näidetena toodud Andmevaatudi vaaded on võetud Andmevaaturi kasutusjuhendist [38],

1. **Analüüside ja uuringute tulemused**: laboratooriumis tehtud uuringute tulemused (nt vere- ja uriiniproovid), füüsilise tervisekontrolli tulemused, elutähtsad tunnused (nt vererõhk, kehatemperatuur jne). Andmevaaturi näide - "Uuringud" vaade ja/või "Antropomeetrilised näitajad" vaade.
2. **Allergiad**: patsiendil esinevad toidu- ja/või ravimiallergiad. Andmevaaturi näide - "Kõrvaltoimed ja allergiad" vaade.
3. **Terviseseisundi ajalugu**: patsiendi elu jooksul kogutud sümptomid ja diagnoosid (nt diabeet, südame probleemid) k.a. pereliikmete terviseseisundi ajalugu. Andmevaaturi näide - "Diagnoosid" vaade ja/või "Patsiendiinfo" vaates "Riski- ja ohutegurid" plokk.
4. **Retseptid**: patsiendile määratud ravimid ja tarbimise juhendid. Andmevaaturi näide - "Retseptid" vaade.
5. **Soovitused**: arsti poolt määratud toitumise soovitused, vajalikud abiseadmed (nt ratastool).
6. **Saatekirjad**: saatekirjad eriarsti vastuvõtule registreerimiseks.

7. **Vastuvõtud**: toimunud vastuvõtude kokkuvõtted. Andmevaaturi näide - "Avavaade" vaates "Ajatelg" ja/või "Patsiendiinfo" vaates "Broneeringud" plokk.
8. **Elustiil**: FHIR Observation ressurssi koodiga *social-history* ja *activity* kirjed. *social-history* koodiga kirjed kirjeldavad patsiendi elukeskkonda ja -harjumusi, mis võivad mõjutada tervist (nt suitsetamine, alkoholi tarbimine). *activity* koodiga kirjed kirjeldavad patsiendi elustiili ja aktiivsust (nt liikumine, toitumine). Andmevaaturi näide - "Patsiendiinfo" vaates "Elustiili andmed" plokk.

3.7 Süsteemi tehniline kirjeldus

PoC rakenduse repositoorium: https://gitlab.cs.ttu.ee/litunt/medical-data-app

Pakutud tehniline lahendus koosneb mitte ainult rakendusest. Tegemist on süsteemiga, milles rakendus omab kasutajaliidese rolli kasutaja ja süsteemi interaktsiooni võimaldamiseks, tegeleb andmete töötlemise ja konverteerimisega ning pakub API-d detsentraliseeritud sisu-adresseeritava salvestusvõrguga integreerimiseks. Rakendus tagab andmete kuvamise ja sisestamise võimalust ning ligipääsu detsentraliseeritud sisu-adresseeritava salvestusvõrku.

Rakendus ja kasutajaliides

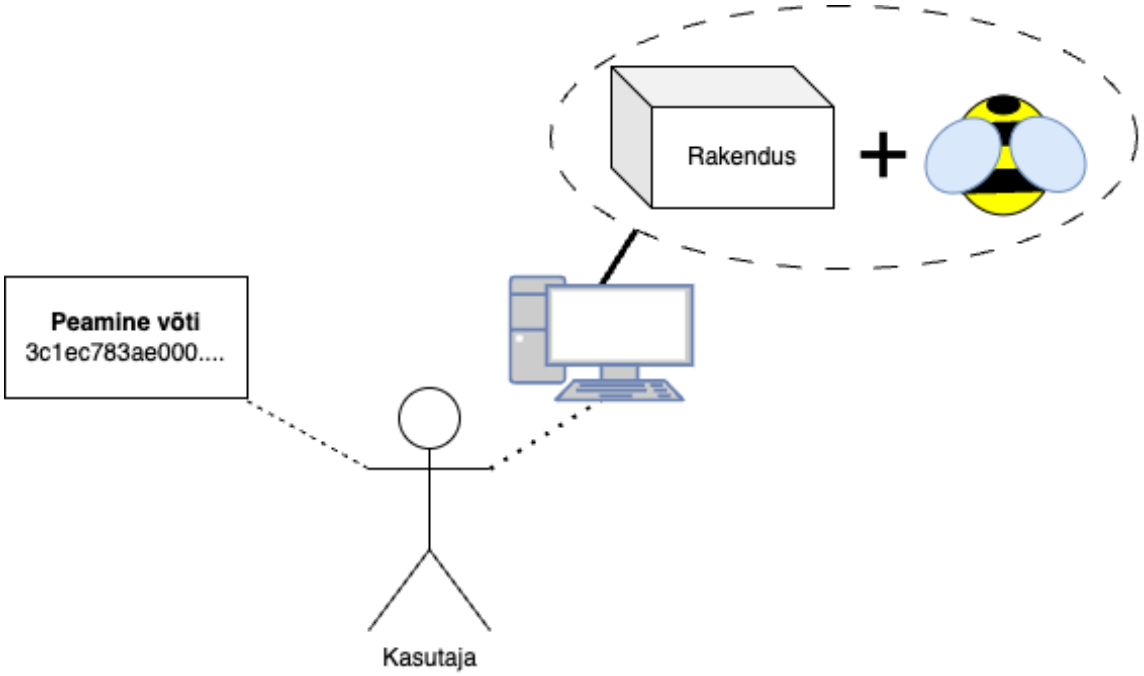
Selleks, et võimaldada rakendusel töötada erinevatel seadmetel ükskõik millise operatsioonisüsteemi peal (Windows, Linux, iOS, macOS), oli valitud C# programmeerimiskeeles põhinev raamistik MAUI [11]. Võrreldes teiste raamistikutega, MAUI peamine eelis on C# programmeerimiskeel. Eestis enamus meditsiinivaldkonna rakendusi on loodud kasutades kas Java või C# programmeerimiskeelt. Seega sellisel juhul on lihtsam leida spetsialiste, kes saaksid arendada ja hooldada uut rakendust. C# programmeerimiskeele jaoks on olemas tõhus *HL7* teek, mis võimaldab lugeda ja koostada FHIR kirjed ning kasutada FHIR ressursse objektidena programmis. Lisaks, on olemas teek nimega *fo-dicom* DICOM failide lugemiseks ja kuvamiseks.

Swarm ja Bee sõlm

Detsentraliseeritud sisu-adresseeritava salvestusvõrguna sai valitud Swarm võrk [13]. Selle eelised teise detsentraliseeritud võrkude üle said põhjalikult selgitatud alapeatükis 2.3 - Ethereum Swarm.

Swarm võrgus osalemiseks kasutajal on vaja paigaldada Bee sõlme [22] enda masinasse. Bee sõlme ühendatakse detsentraliseeritud võrgu teiste sõlmedega. Sõlm ei ole isikus-

tatud ega konkreetse kasutajaga seotud. Bee sõlme olemasolu masinas tagab ligipääsu detsentraliseeritud sisu-adresseeritava salvestusvõrku. Kasutaja vaatenurgast ei ole vahet, millise sõlme pealt toimub interaktsioon Swarm detsentraliseeritud sisu-adresseeritava salvestusvõrguga. Swarm võrgustikku ligipääsemiseks peab kasutajal olema enda isiklik privaatne peamine võti, mis tagab ligipääsu enda ja ainult enda andmete komplektile (Joonis 18).



Joonis 18. *Kasutaja interaktsioon rakenduse ja Bee sõlmega*

Andmete salvestamiseks Swarm võrgus peab olema Bee sõlme peal kehtiv postmark ehk *Postage Stamp*. Andmete üleslaadimisel postmark "kinnitatakse" iga andmetükki külge. Postmargi eluiga (*batch TTL*) määrab, kui oluline on antud andmetükki säilimine ning kui pikalt peab antud andmetükk olema süsteemis säilinud. Postmargi kasutamine tagab Swarm salvestusruumi kasutamise võimalust. Kui Postmargi kehtivusaeg on lõppenud, lõpeb ka sõlme jaoks kohustus andmetükki säilimiseks ning andmetükk kustutatakse süsteemist ära [39]. Tegemist on *Garbage Collector* protsessi ühe osaga.

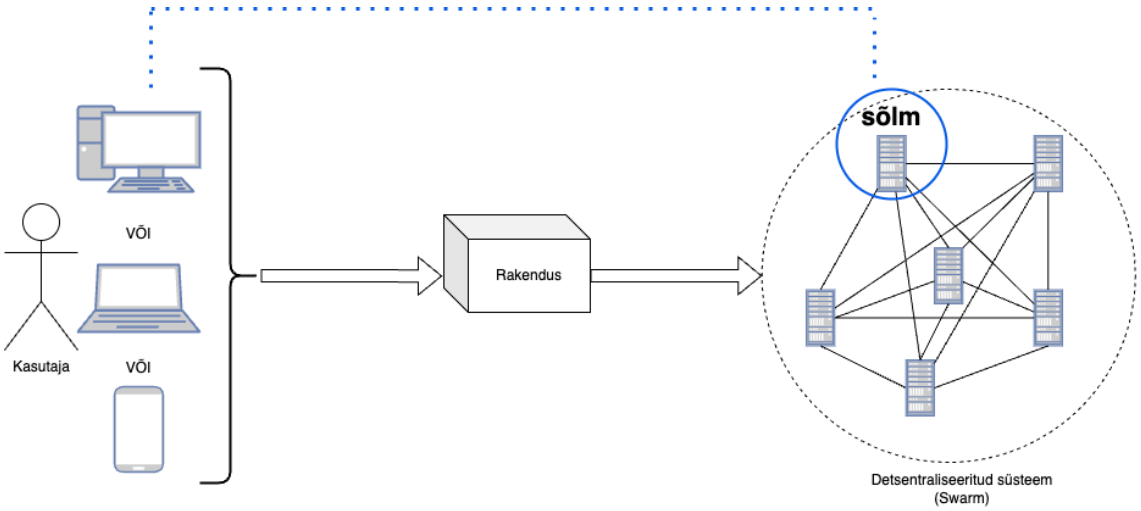
3.8 Süsteemi arhitektuur

Süsteem tervikuna koosneb kolmest osast:

1. **detsentraliseeritud sisu-adresseeritav salvestusvõrk (DCAS)**, kus andmed on hoitud;
2. **sõlm**, mis on tegelikult osa detsentraliseeritud süsteemist ja paikneb kasutaja

seadmes;

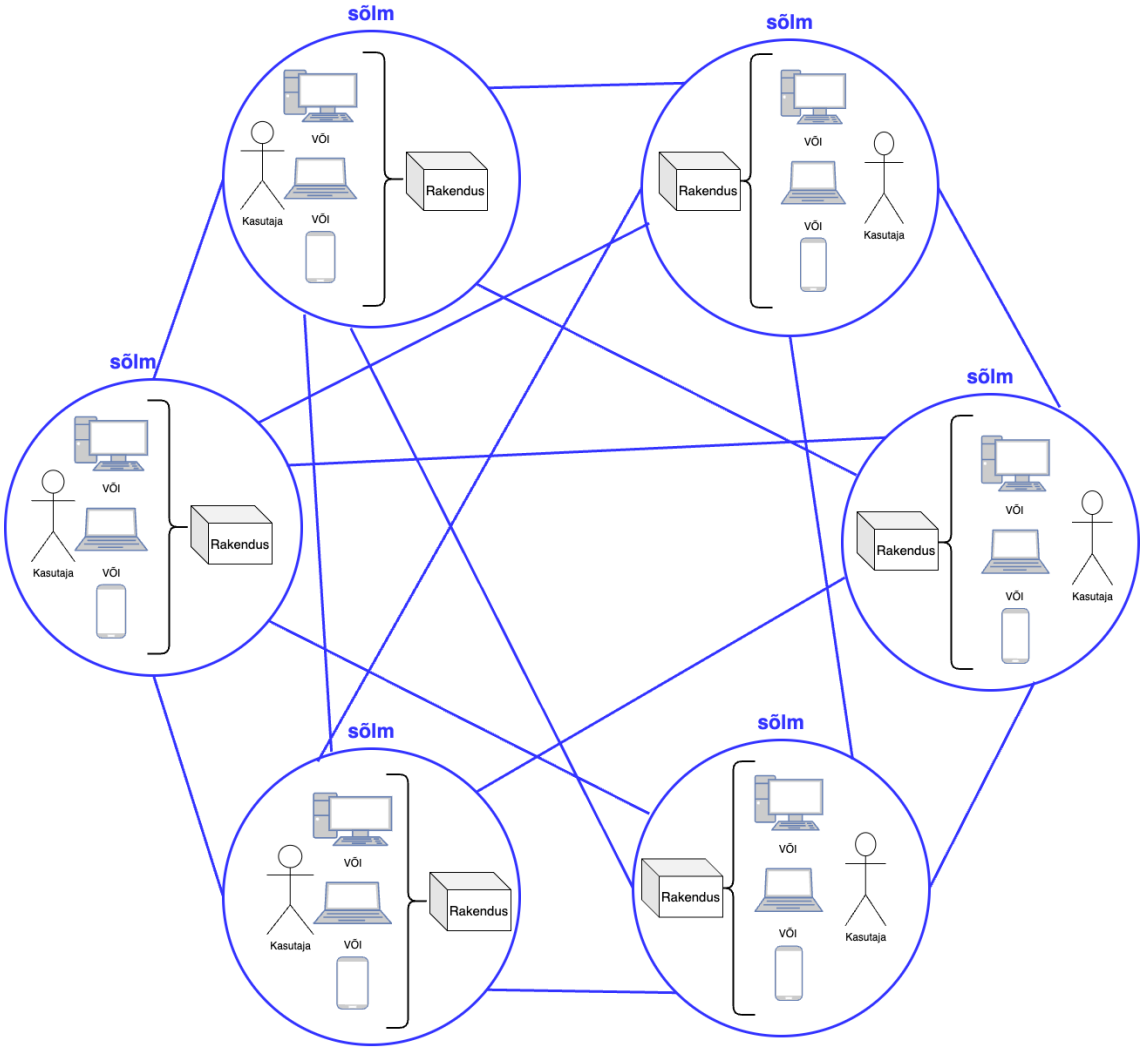
3. **rakendus**, mis pakub kasutajaliidest ja vajalikku funktsionaalsust andmete töötlemiseks selleks, et andmed salvestada detsentraliseeritud sisu-adresseeritavas salvestusvõrgus ning pärida andmed DCAS võrgust tagasi.



Joonis 19. *Kogu süsteemi (rakendus + DCAS) arhitektuur*

Joonisel 19 näidatakse süsteemi arhitektuuri komponentide vahelised seosed - kasutaja interaktsioon kogu süsteemiga tema seadme kaudu. Seadmesse on installeeritud Bee sõlm, mis tagab ligipääsu DCAS süsteemi. Kasutaja ja detsentraliseeritud sisu-adresseeritava süsteemi vaheline interaktsioon toimub rakenduse kaudu.

Joonisel 19 näidatakse, et kasutajal ei ole otseset interaktsiooni detsentraliseeritud süsteemiga, kõik toimingud toimuvad vahekihilise rakenduse kaudu.



Joonis 20. *Süsteemi detsentraliseerimise printsiibi illustreerimine*

Detsentraliseeritud sisu-adresseeritavas salvestusvõrgus osalemine eeldab Bee sõlme olemasolu seadmes. Järelikult, Bee sõlme ja rakenduse seost võib kirjeldada selliselt, et seade koos sinna installeeritud rakendusega on üks terviklik osa detsentraliseeritud sisu-adresseeritavast süsteemist, mis tagab nii ligipääsu DCAS süsteemi (Bee sõlm), kui ka võimaldab DCAS süsteemis hoitud andmetega interaktsiooni ja andmete töötlemist (rakendus). See on illustreeritud Joonisel 20. Joonisel 20 selgitatakse, et seade ja rakendus mitte ei kasuta DCAS süsteemi, vaid moodustavad detsentraliseeritud sisu-adresseeritava salvestusvõrgu instantsi. Detsentraliseeritud sisu-adresseeritavas salvestusvõrgus osalemiseks ei ole oluline, mille seadme kaudu üritab kasutaja ligipääsu saada. Oluline on see, et seadmesse on installeeritud Bee sõlm - see juba tagab ligipääsu DCAS süsteemi. Järelikult see tähendab seda, et detsentraliseeritud sisu-adresseeritava salvestusvõrku ligipääs on olemas sellest seadmest, millele on Bee sõlm installeeritud.

Bee sõlm süsteemi arhitektuuris

Kasutaja seadmes Bee sõlme instantsi installeerimine ja käivitamine tagab ligipääsu detsentraliseeritud sisu-adresseeritava salvestusvõrku. Kuigi Bee sõlm on installeeritud kasutaja seadmes, sõlm ise on osa detsentraliseeritud süsteemist (Swarm). Bee sõlme töö põhimõtete detailne kirjeldus on alapeatükis 3.7 (Bee sõlm).

DCAS süsteemi arhitektuuris

Detsentraliseeritud sisu-adresseeritav salvestusvõrk omab kirjeldatud süsteemis andmete detsentraliseeritud salvestusruumi rolli. Detsentraliseeritud sisu-adresseeritava salvestusvõrgu töö põhimõtted, k.a. andmete hoidmine ja küsimine on kirjeldatud alapeatükkides 2.2, 2.3 (Ethereum Swarm) ja 3.4 (Andmetele ligipääsu mehhanism ja turvalisus).

Rakendus süsteemi arhitektuuris

Rakendus on süsteemi keskne komponent, mis ühendab kasutajaliidese, andmete valideerimise, serialiseerimise ning FHIR-andmevormingute halduse. Selle eesmärk on võimaldada nii lõppkasutajatele kui ka süsteemidele andmetega opereerimist. Andmetega opereerimine peab olema paindlik, reeglipõhine ja standardipõhine. Rakenduse võimekus kasutada mitmeid FHIR-formaate annab sellele tugeva eelise erinevate sidus- ja partnerplatvormide integreerimisel ning võimaldab rohkem variante ja valikuid lõpliku andmete vormingu otsustamisel.

Detailselt on rakenduse funktsionaalsus kirjeldatud alapeatükis 3.9.

3.9 Rakenduse struktuur ja disain

Rakenduse roll süsteemis on andmetöötlus ja kasutajaliides ehk süsteemi ja kasutaja vahelise interaktsiooni tagamine. Rakendus täidab kirjeldatud süsteemis mitu funktsiooni:

- kasutajaliidese vormistamine,
- kasutaja poolt sisestatud andmete valideerimine ja töötlemine,
- DCAS süsteemist saadud andmete mappimine,
- FHIR-andmevormingute haldamine ja konverteerimine.

Kasutajaliides

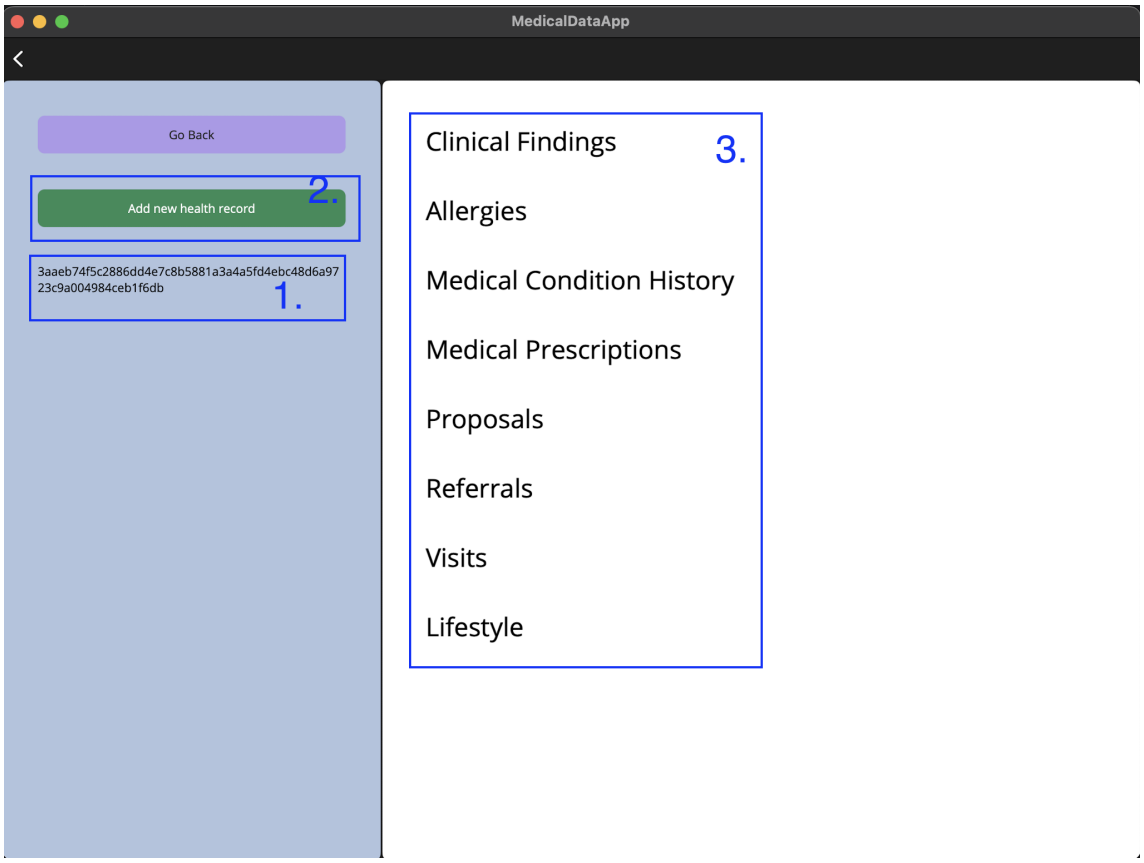
Rakendus pakub kasutajale interaktiivset kasutajaliidest. Kasutajaliidese kaudu saab:

- terviseandmed sisestada,
- varem salvestatud andmed vaadata.

Kasutajaliidese kaudu näeb kasutaja alapeatükis 3.6 kirjeldatud FHIR terviseandmete kategooriad. Valitud kategooriale klikkides näeb kasutaja valitud kategooria alla kuuluvad terviseandmed, mis olid varem sisestatud ning kuuluvad kasutaja terviseandmete komplekti.

Joonis 21 on ekraanitõmmis rakenduse pealehelt. Pealehele jõuab kasutaja siis, kui:

- on sisenenud rakendusse terviseandmete peamise võtmega või
- on valinud uue terviseandmete struktuuri loomist.

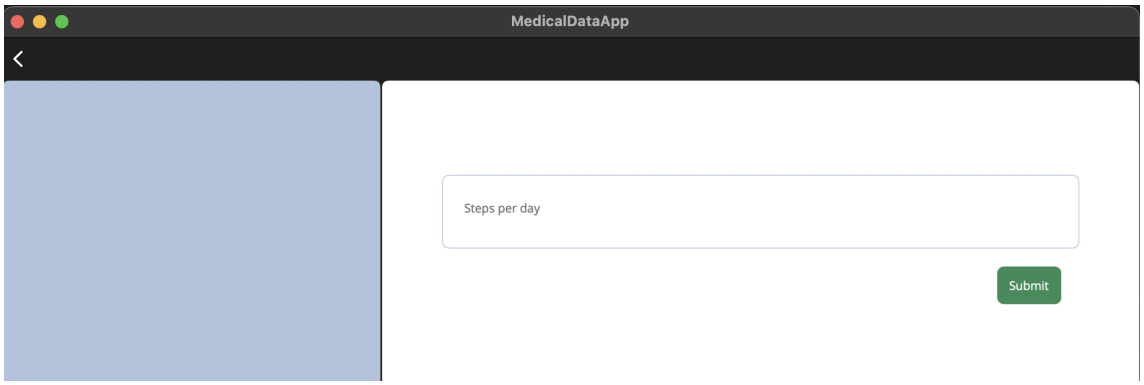


Joonis 21. *Rakenduse pealehe vaade*

Punkt nr 1: Esimese olukorra puhul (kasutaja on rakendusse sisenenud peamise võtmega) näeb kasutaja pealehel selle sama peamise võtme väärtust. Teise olukorra puhul (kasutaja on valinud uue terviseandmete struktuuri loomist) näeb kasutaja pealehel tühja andmete komplekti peamist võtit. Tühja andmete komplekti peamine võti on kättesaadav siis, kui rakendus salvestab DCAS süsteemis tühja põhistruktuuri ning saab DCAS süsteemist päringu vastuseks viidet. Peamise võtme kuvamine oli rakenduses arendatud praeguses

etapis testimise mugavuse tagamiseks.

Punkt nr 2: Nupp uue terviseandmete kirje loomiseks. Suunab kasutajat terviseandmete lisamise kategooria valimise lehele. Vajutades "Add new health record" nupule, suunatakse kasutajat terviseandmete sisestamise kategooria valimise vaatele. Terviseandmete sisestamiseks valib kasutaja soovitud kategooriat. Kui kategoorial on ka alamkategooriaid, siis suunatakse kasutajat alamkategooriate valimise vaatele (vt Lisa 15). Vastavalt valitud (alam)kategooriale avaneb vaade vajalikke sisendväljadega (Joonis 22).



Joonis 22. *Rakenduses terviseandmete sisestamise vaade*

Rakenduse kaudu sisestatud terviseandmed edastatakse andmetöötlusmoodulisse. Andmetöötlusmoodul vastutab sisestatud terviseandmete valideerimise, konverteerimise ja sobivaks kujuks vormistamise eest.

Punkt nr 3: FHIR terviseandmete kategooriad. Kategoriseerimine on detailselt esitatud alapeatükis 3.6. Terviseandmete vaatamiseks valib kasutaja kategooriat. Valitud kategooria all kuvatakse kategooria alla koondatud terviseandmete viidete indeks(id). Indeksi sisse on peidetud nimekiri terviseandmete FHIR ressursside viidetest. Järgmise vaatena kuvatakse kasutajale nimekiri FHIR ressurssidest lühendatud kujul (vt Lisa 16).

Terviseandmete sisestamiseks vaadete genereerimine

Vaated terviseandmete sisestamiseks genereeritakse rakenduses dünaamiliselt. Dünaamiline kujundamine (*dynamic rendering*), üldistamine (*generic*) ning *factory* on antud juhul valitud koodi disaini mustrid. Eelnimetatud koodi disaini valikud olid tehtud järgmistel põhjustel:

- Praegu on olemas 8 FHIR terviseandmete kategooriat, igas kategoorias on vähemalt 1 andmete tüüp olemas (nt vererõhk, südamelöögi sagedus). Järelikult, ilma koodi

struktuuri üldistamata peab projektis olema eraldiseisev vaade iga kategooria alla kuuluva FHIR andmete tüübi jaoks;
- Juhul, kui rakenduse kujundus peaks muutma, mitme vaate puhul peab muudatused tegema kõikides vaadetes. Dünaamiliselt ehk koodi abil genereeritud kujundusega selliseid muudatusi tegema ei pea;
- FHIR terviseandmete sisendväljad on kõikide andmete tüüpide puhul kujunduse poolest samad. Erinevad võivad olla vaid, näiteks, sisendvälja pealkiri ja *placeholder*. Mõnikord sõltuvalt andmetüübist võib valida ka teistsuguse sisendvälja tüübi (nt tekstiväli, numbriväli jne). Sellise lähteülesande puhul oli mõistlik valida *generic* lähenemist.

Terviseandmete sisestamise vaate kujundus põhineb valitud kategoorial. Vaate kujundamine initsialiseeritakse terviseandmete sisestamise vaate valimise hetkel (vt Lisa 8). Terviseandmete sisestamise vaate ehitamine toimub klassis *HealthDataViewFactory* (vt Lisa 9). Klassis *HealthDataViewFactory* ehitatakse terviseandmete sisestamise vaade (*HealthDataView*) vajalikke parameetrite ja konfiguratsioonidega vastavalt etteantud kategooriale. Kui rakenduses tekib vajadus võimaldada sisestada teistest kategooriatest ja tüüpidest terviseandmed, siis selle jaoks tuleb täiendada *HealthDataViewFactory* klassi uue kategooriaga ja selle kujundusele vajalikke parameetritega.

Andmetöötlus ja konverteerimine

Sisestatud terviseandmete kättesaamise implementeerimine on kujundatud Lisas 10.

Samm 1: Kasutajaliideses kuvatakse varem dünaamiliselt genereeritud andmete sisestamise vaade. Selle vaate kaudu sisestab kasutaja terviseandmed. Andmed on edastatud edasi rakendusse.

Samm 2: Valitakse vastavalt terviseandmete tüübile andmete töötlemise protsessor. Protsessori valik tehakse vastaval terviseandmete tüübile (nt vererõhk, sammud, südamelöögi sagedus jne). Kasutaja poolt sisestatud väärtused suunatakse edasi andmete töötlemiseks ja serialiseerimiseks valitud protsessorisse.

Samm 3: Terviseandmete töötlemise ja serialiseerimise tulemuseks on FHIR ressurss, mis JSON formaadis saadetakse edasi salvestamiseks.

Terviseandmete töötlemise ja serialiseerimise protsessor valitakse *Factory Method* [40] disainimustri järgi vastavalt etteantud terviseandmete tüübile (vt Lisa 11). Iga terviseandmete protsessor (näiteks, südamelöögi sageduse andmete protsessor ehk *HeartRateDataPro-*

cessor, vt Lisa 12) implementeerib protsessori liidese (vt Lisa 13). Üldistatud liidese implementeerimine aitab säilida ühtset stiili ja struktuuri.

FHIR ressursside mallid on hoitud rakenduses eraldi *template*-failidena. Iga FHIR terviseandmete tüübi jaoks on oma malli fail. Mall koosneb FHIR struktuurist ning sisaldab ressurssi kirjeldatavaid andmeid. Mallist võetud struktuur täiendatakse sisestatud terviseandmete väärtustega. Saadud struktuurist ehitatakse lõplik FHIR ressurss.

Kirjeldatud andmete konverteerimine ja serialiseerimine toimub FHIR JSON formaadis. FHIR Turtle ehk RDF formaadis andmete töötlemise prototüüp on arendatud eraldi Git harus [41] klassis *FhirRdfParser* ning selle kasutamine on implementeeritud klassis *FhirRdfMapper*.

Mitme FHIR-andmeformaadi tugi

Rakendusel on võimekus töötada erinevate FHIR-serialiseerimise vormingutega, sealhulgas:

- FHIR JSON – enim kasutatav formaat REST API-de puhul; masinloetav ja hästi dokumenteeritud. Kasutatakse käesoleva töö raames andmetega opereerimisel;
- FHIR Turtle (TTL) – RDF-põhine formaat, mis sobib hästi semantilise veebi ja ontoloogiapõhise andmetöötlusega.

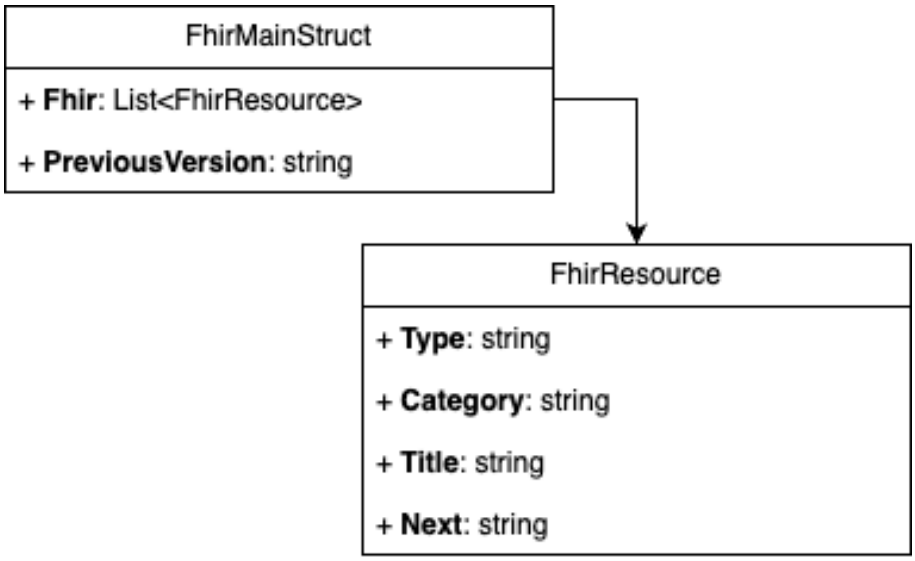
Tänu mitmekülgsele formaaditoele rakendust on võimalik igal hetkel kohandada vastavalt muutuvatele nõuetele, kuna praegusel hetkel terviseandmete vorming ei ole veel lõplikult valitud.

Lisaks, lõputöö käigus oli uuritud FHIR Shorthand formaadi implementeerimise võimalus. FHIR Shorthand (FSH) on domeenispetsiifiline keel, mida kasutatakse FHIR-profiilide loomiseks. Võrreldes FHIR JSON formaadiga, FSH formaadi jaoks ei ole olemas C# programmeerimiskeeles sobivat teeki. FHIR Shorthand formaadi töötlemiseks on olemas *SUSHI* (FSH $\rightarrow$ JSON) ja *GoFSH* (JSON $\rightarrow$ FSH) teegid, mis on arendatud JavaScript ja TypeScript keeltes. Seega oleks võimalik arendada eraldiseisvat Node.js mikroteenust, mille abil saaks FSH-failidest genereerida vastavaid JSON-struktuure ja vastupidi. SUSHI [42] ja GoFSH [43] teekide töövõimekust on võimalik valideerida *fshonline* [44] tööriista abil.

3.10 Terviseandmete käsitlemine rakenduses

Tekstikujuliste terviseandmete struktuur

Käesoleva lõputöö raames arendatud rakenduse jaoks FHIR andmetega opereerimiseks oli valitud JSON formaat. Struktuuri representeerimiseks rakenduses koodi sees on kaks klassi: *FhirMainStruct* ja *FhirResource*. *FhirMainStruct* kasutatakse terviseandmete kõige peamise struktuuri esitamiseks. Nimekiri *FhirResource* instantsidest peamises struktuuris sisaldab nimekirja FHIR terviseandmete kategooriatest (Joonis 23). Esitatud FHIR terviseandmete kategooriad on need kategooriad, mis peegeldavad FHIR ressursside jaotamist terviseandmete kategooriateks (vt 3.6). Praeguses arekdnuse aenduse etapis nimekiri terviseandmete kategooriatest luuakse automaatselt koos esialgse andmekomplekti struktuuri loomisega. Esialgse andmekomplekti struktuuri antud juhul esitatakse *FhirMainStruct* klassi abil.



Joonis 23. *FHIR terviseandmete mudelid rakenduses*

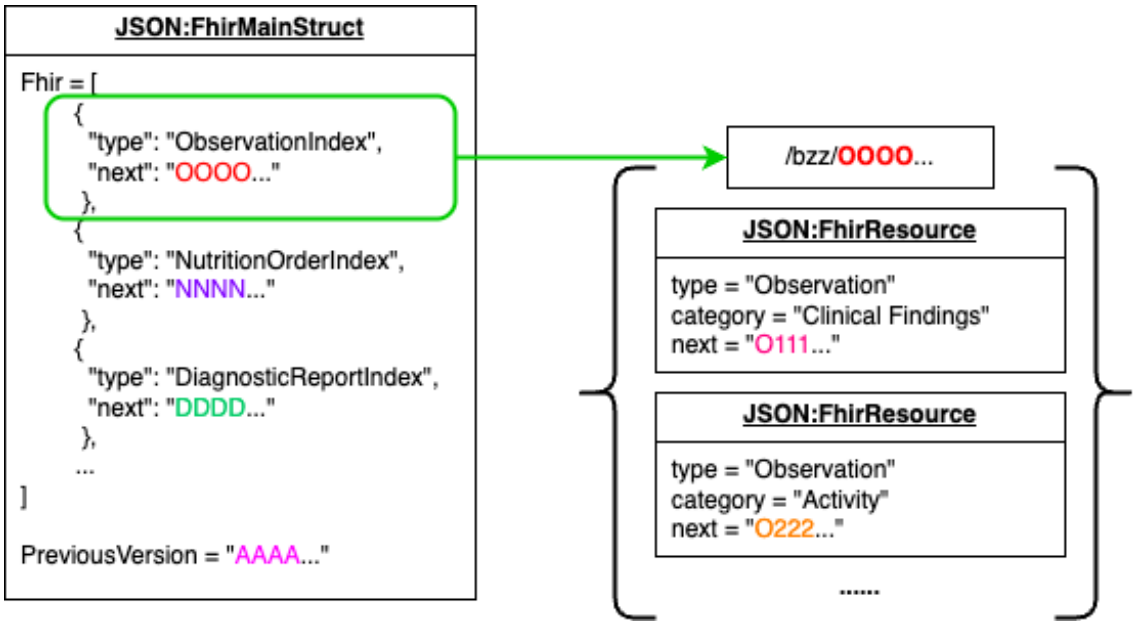
FhirResource tüübi instantsid võivad olla esitatud nii üksikute, kui ka mitmete andmete osadena. JSON formaadis see tähendab vastavalt kas üksikut JSON andmetükki või JSON Array ehk JSON nimekirjana representeeritud andmetükki.

Tekstikujuliste terviseandmete laadimine DCAS süsteemist

Terviseandmed DCAS süsteemist laetakse osadena vastavalt vajadusele. Osaliselt andmete laadimise printsiibina sai valitud *Load on Demand* ehk vajaduspõhine laadimine või *Lazy Loading* [45]. Andmete vajaduspõhine laadimine võimaldab hajutada ja vähendada

teatud ajahetkel andmete laadimise koormust ning vältida ebavajalikku ligipääsu tervele terviseandmete komplektile üheaegselt.

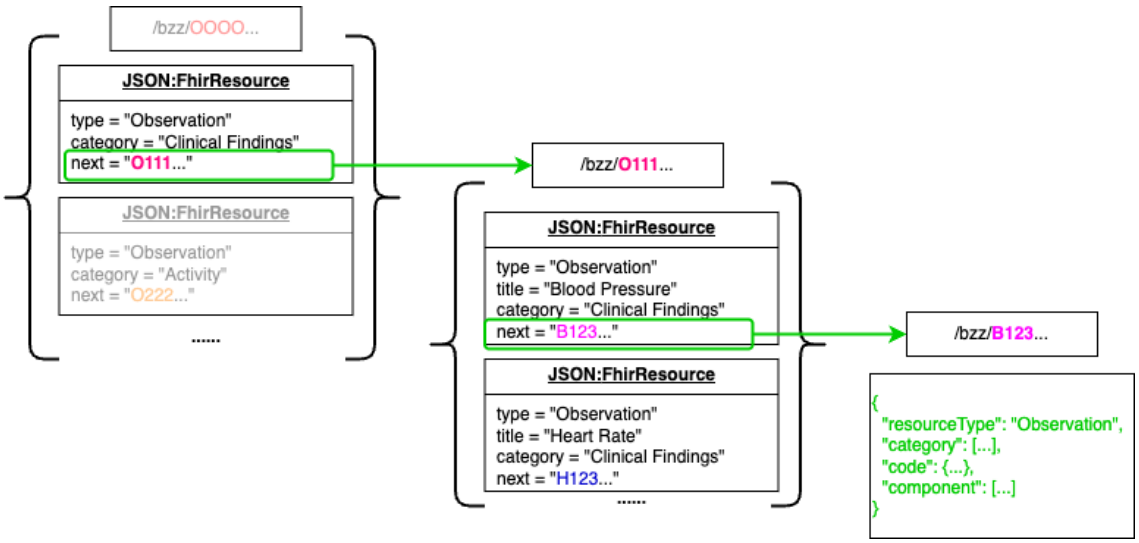
Kategooria valimisel *FhirMainStruct* struktuurist võtab programm räsiväärtuse formaadis viidet (*next* väärtust) järgmisele andmete osale. Järgmine andmetükk on nimekiri *FhirResource* instantsidest, mis kuuluvad valitud kategooria alla (Joonis 24). Need instantsid on terviseandmete indeksid, mis representeerivad terviseandmete alamkategooriate nimekirju. Joonisel 24 on näitena valitud nimekiri Observation tüübi FHIR ressurssidest ehk alamkategooriatest. Täpsemalt FHIR ressursside kategoriseerimine on kirjeldatud alapeatükis 3.6. Programm võtab kasutaja poolt valitud FHIR terviseandmete kategooria *next* väärtuse (`"OOOO..."`). Väärtus kasutatakse `GET /bzz` päringus viite (*reference*) väärtusena (vt 3.4). Programm teeb päringu DCAS süsteemi. DCAS süsteemist saadud vastusest laetakse nimekiri alamkategooriatest. Valitud alamkategooriad kuuluvad konkreetselt valitud Observation tüübi FHIR ressursside alla (vt 3.6). Joonisel 24 on näitena toodud *Clinical Findings* ja *Activity* alamkategooriad. Laetud alamkategooriad kuvatakse rakenduse kasutajaliideses. Edaspidi andmete laadimine toimub samamoodi vajaduspõhiselt. Kasutaja ise algatab andmete laadimise protsessi valides soovitud terviseandmete kirje selle sisu kuvamiseks või terviseandmete kategooriate navigeerimise jätkamiseks.



Joonis 24. *FHIR terviseandmete sõltuvuste mudel*

Joonisel 25 näidatakse järgmised sammud peale alamkategooria valimist. Programm võtab kasutaja poolt valitud FHIR terviseandmete alamkategooria *next* väärtuse (`"O111..."`). Väärtus kasutatakse `GET /bzz` päringus viite (*reference*) väärtusena (vt 3.4). Programm teeb päringu DCAS süsteemi. DCAS süsteemist saadud vastusest laetakse nimekiri valitud

alamkategooriasse kuuluvatest FHIR ressursside viidetest. Terviseandmete alamkategooriasse kuuluvad FHIR ressursside viited esitatakse kasutajaliideses kui terviseandmete pealkirju. Antud hetkel need andmed ei ole veel täielikult laetud FHIR ressursside andmed. Selleks, et laadida tervikliku FHIR ressurssi, valitakse soovitud terviseandmete kirje lühendatud representatsioon. FHIR ressurssi lühendatud representatsioon sisaldab viidet terviklikule FHIR ressurssi sisule oma *next* väärtuses (`"B123..."`). Väärtust kasutatakse `GET /bzz` päringus viite (*reference*) väärtusena (vt 3.4). Programm teeb päringu DCAS süsteemi. DCAS süsteemist saadud vastusest laetakse FHIR ressurssi JSON formaadis andmed. Edaspidi jätkub rakenduses andmetöötluse protsess.



Joonis 25. *FHIR terviseandmete üksikute ressursside sõltuvuste mudel*

Tekstikujuliste terviseandmete serialiseerimine

Peamine andmete töötlemise töövoog toimub rakenduses JSON formaadis. FHIR ressurssi objekti JSON formaadiks ja vastupidi JSON formaadist objektiks konverteerimiseks kasutatakse *FhirEntryJsonConverter* klass (vt Lisa 14). FHIR andmete jaoks kohandatud konverter kasutatakse:

- terviseandmete protsessorites, kus võetakse FHIR terviseandmed JSON formaadis malli failist ning konverteeritakse programmi objektideks;
- enne terviseandmete salvestamist DCAS süsteemis - FHIR ressursside programmi objektid konverteeritakse JSON formaadiks;
- terviseandmete pärimisel DCAS süsteemist - FHIR terviseandmed JSON formaadis konverteeritakse programmi objektideks edaspidiseks kasutamiseks rakenduse loogikas.

DICOM pildifaili salvestamine ja laadimine DCAS süsteemist

DICOM pildiandmete DCAS süsteemi salvestamise valideerimiseks oli käesoleva töö raames valitud DICOM test-fail avalikust DICOM failide kogust DICOM Library [46]. DICOM fail oli juba eelnevalt rakenduse sisse laetud. See asendab rakenduse prototüübi tasemel faili päris laadimise protsessi kasutaja seadmest rakendusse. Käesoleva töö raames arendatud DICOM pildifaili salvestamise funktsionaalsuse eesmärk oli valideerida, kas DICOM pildiandmede salvestamine DCAS süsteemi õnnestub. Rakenduses selleks teostatakse järgmiseid samme (vt Lisa 17):

Samm 1: Faili lugemine rakenduse sisse ning konverteerimine baitide massiiviks - `byte[]`.

Samm 2: Faili baitide salvestamine DCAS süsteemi (vt 3.5 - DICOM pildiandmete salvestamine) `POST /bytes` päringu abil.

Samm 3: Faili baitide laadimine DCAS süsteemist rakendusse (vt 3.5 - DICOM pildiandmete salvestamine) `GET /bytes` päringu abil. Päringu vastusena saadud faili baitid salvestatakse `.dcm` laiendusega uue failina rakenduses. Uue faili on võimalik pärast vaadata.

3.11 Süsteemi testimine

Käesoleva töö raames rakenduses olid testitud selle kõige peamised protsessid. Suurem osa kogu süsteemist oli testitud manuaalselt. Manuaaltestimise võimaldamine oli üks põhjustest, miks üleslaetavate FHIR andmete formaadiks praeguses etapis (rakenduse valmidusaste TRL 3) sai valitud JSON - loetavuse ja andmete kättesaamise mugavuse pärast. Rakenduse üksikud funktsionaalsuse osad on lisaks kaetud ka ühik- ja integratsioontestidega.

Osa tehnilise teostatavuse valideerimisest käesoleva töö raames oli näidata ka seda, et valitud detsentraliseeritud sisu-adresseeritav salvestusvõrk Swarm on võimeline töötama ilma tõrgeteta suurema koormuse korral - veenduda, et süsteem on võimeline töötlema palju päringuid korraga ning päringu vastusest saadud andmete kvaliteet ei ole halvenenud, k.a. pildiandmete kvaliteet.

Manuaaltestimine

Manuaalselt kõige rohkem oli vaja testida andmete sisu ja säilinud kvaliteedi juhul, kui

andmed olid salvestatud detsentraliseeritud sisu-adresseeritavas salvestusvõrgus ning küsitud kuvamiseks.

DCAS süsteemi testimise eesmärk oli kontrollida, kas andmed salvestatakse korrektselt ning kas need on hiljem süsteemist leitavad ja loetavad. Peamine test-stsenaarium valideerimiseks oli andmete kättesaamine ja kuvamine. Kõigepealt, terviseandmed olid salvestatud DCAS süsteemis.

Terviseandmete sisu valideerimine: Andmete sisu vaatlemiseks oli kasutatud kas veebilehitsejat või cURL tööriista. Selle abil oli võimalik valideerida tervikliku JSON formaadis salvestatud terviseandmete seisu.

Terviseandmete kuvamise valideerimine: Andmete kuvamise korrektsus oli valideeritud arendatud rakenduse kasutajaliideses. Ülesandeks oli veenduda, et kasutajale kuvatakse vajalik terviseandmete sisu ettenähtud kujul.

Ühik- ja integratsioontestid

Praeguses rakenduse valmimise etapis automaattestidega oli kaetud Swarm süsteemi teenus - *SwarmService* klass. Antud juhul Swarm süsteemiga integratsioon on kõige kriitilisem funktsionaalsus ning selle stabiilsus peab olema tagatud. Teised rakenduse osad (nt FHIR ressursside mallide lugemine, terviseandmete sisestamise vaadete dünaamiline genereerimine jne) olid testitud peamiselt manuaalselt. Lisaks, autor arvestas ka sellega, et praeguses etapis arendatud rakendus on esialgne prototüüp, mis võib edaspidi oluliselt muutuda. Seega ainuke põhifunktsionaalsus, mis ei tohiks muutuda, on integratsioon Swarm süsteemiga.

Koormustestid

Lõputöös koormustestide läbiviimiseks oli kasutatud Jmeter tööriist [47]. Jmeter tööriistaga oli testitud detsentraliseeritud sisu-adresseeritava salvestusvõrgu Swarm võimekus töötada suurema koormuse puhul. Testi ehitamisel oli valitud eelnevalt üleslaetud andmete viide ning sellega oli koostatud päring Swarm süsteemi. Testi tingimused olid katsetatud erinevate kombinatsioonidega:

- päringute N arv sekundis
- päringut tehtavate kasutajate N arv sekundis

Lisaks, koormustesti tulemusena oli vaja veenduda, et saadud vastuses andmed on samasu-

gused kõikide tehtud päringute puhul, ning andmete kvaliteet on säilinud. Jmeter tööristaga oli näitena genereeritud raport koormuse testimise situatsioonile, kui 1000 samaaegselt üht ja sama päringut tehakse 1000 korda paralleelselt (vt Lisa 2).

4. Järeldused ja analüüs

Käesolevas peatükis vaadeldakse ja valideeritakse lõputöö tulemusi vastavalt püstitatud eesmärgile, tehakse tulemuste põhjal järeldused, kirjeldatakse tuleviku uurimis- ja edasiarengu suundasid ning analüüsitakse lõputöö teostamise protsessi ja tehnoloogilisi valikuid.

Lõputöö eesmärgiks oli analüüsida, kavandada kasutusjuhud ja arendada API-d selleks, et valideerida detsentraliseeritud sisu-adresseeritavas salvestusvõrgus terviseandmete hoidmise ja kasutamise tehnilist teostatavust ja realiseerida TRL 3 tasemel [7] prototüüp selleks, et saaks hakata valideerima tehnoloogia kasutatavust kõrgematel TRL tasemetel.

4.1 Tehtud töö paiknemine suure uurimisprojekti kontekstis

Käesoleva lõputöö raames tõstetatud idee isiklike terviseandmete detsentraliseeritud sisu-adresseeritavas salvestusvõrgus hoidmisest ja kasutamisest esmast kasutamist haiguste diagnoosimisel ja ravil valideeritakse kuni TRL 6 valmidusastme tasemeni TemTA 105 [48] ja PGR 2629 [49] projektide raames, milles eMedLab on osaline.

Mõnede teemade näited, millega suure projekti raames tegeletakse: FHIR terviseandmete tõlgendamine ja töötlemine ning vastava kasutajaliidese dünaamiliselt genereerimine, terviseandmetega opereerimiseks kasutajasõbraliku ja intuitiivse kasutajaliidese disaini väljatöötamine.

Käesoleva lõputöö raames oli katsetatud, kuidas detsentraliseeritud sisu-adresseeritavas salvestusvõrgus võiks toimida terviseandmete salvestamine ja millisel kujul. Oli läbi proovitud, kuidas salvestada teksti- ja pildiandmed. Käesoleva lõputöö raames käsitletud projekti töö fookusest jäi välja geeniandmete salvestamine detsentraliseeritud sisu-adresseeritavas salvestusvõrgus.

4.2 Terviseandmete kasutamine

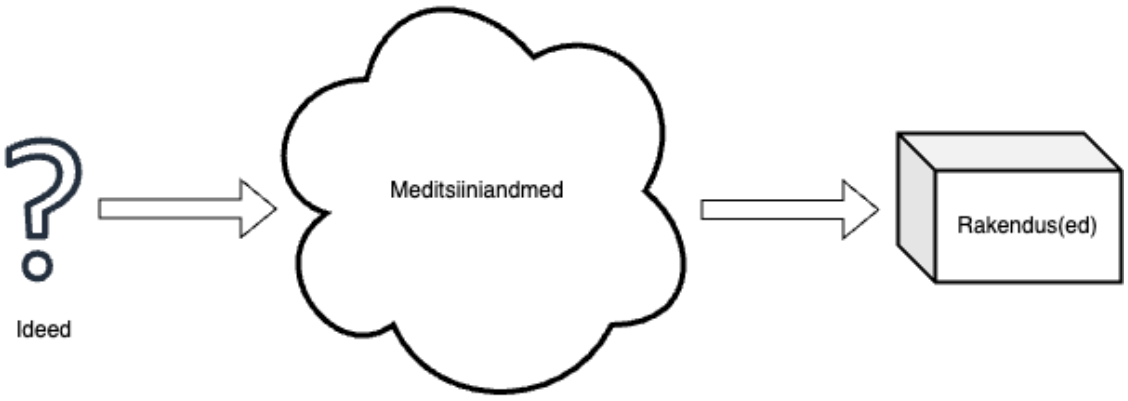
Andmemaht maailmas kasvab eksponentsiaalselt, igapäevaselt luuakse ja talletatakse üha suuremaid andmehulkasid [50] [51] [52]. Terviseandmete maht moodustab tänasel päeval juba 30% kogu andmetest maailmas ning kasvab kiiremini kui teistes valdkondades [53]. Aastaks 2025 prognoositakse kasvu kuni 36%. Samal ajal, ligi 97% nendest meditsiiniand-

metest jääb kasutamata [53]. Lisaks andmete mahule, tegemist on ka rahakuluga. 2020. aastal moodustasid tervishoiukulud Eesti riigi sisemajanduse koguproduktist 7,8% [11].

Tervishoiu valdkonnas infotehnoloogia juurutamisega kasvab andmete hulk digitaalsel kujul [54]. Eriti isiklikest terviseandmetest rääkides on soov saavutada justkui vastupidiseid eesmärke – andmete kaitstus ja andmete kättesaadavus. Isiklikud terviseandmed on ülimalt kasulikud ning võib julgelt öelda, et suur potentsiaal jääb praegu veel kasutamata just isiklike terviseandmete kättesaamatuse tõttu [54].

Nimetatud isiklike terviseandmete mõiste alla kuuluvad ka isiklikkel nutiseadmetel hoitud andmed. Näiteks, nutikellad, mis pidevalt koguvad infot inimese tervisest reaalajas; fitnessäpid, mis samamoodi kas iseseisvalt jälgivad inimese terviseseisundit või inimene ise kannab infot rakenduse sisse. Need jooksvad ja pidevalt kogutud andmed inimese tervisest omaksid suurt väärtust tervishoiu tehnoloogiate arengus [55].

Joonis 26 illustreerib meditsiiniandmete puudust erinevate andmete kasutamise ja rakendamise võimaluste puhul.



Joonis 26. *Meditsiiniandmete puudus potentsiaalsete kasutamise võimaluste jaoks*

4.3 Tervishoiusüsteemide areng

Eesti näitel, esialgu terviseandmed olid hoitud paberkandjal patsiendikaartidena. Need patsiendikaardid säilitati vastavates polikliinikutes ja/või haiglates, kus patsient oli registreeritud. Patsiendil oli ligipääs nendele andmetele ainult siis, kui ta pöördus otse arsti poole. Iga tervishoiuasutus töötles andmeid paberkandjal, mis asusid konkreetses asutuses. Visiidiaja broneerimiseks tuli samuti kas kohale minna või helistada konkreetsesse tervishoiuasutusse. Ühtne süsteem puudus. Seda olukorda võib pidada isiklike terviseandmetega opereerimise esimeseks etapiks.

Teine etapp on isiklike terviseandmete hoidmine ja kasutamine digitaalsel kujul. Eestis algas see protsess 2008. aastal, kui käivitati tervise infosüsteem ja patsiendiportaal Digilugu (nüüd Terviseportaal) [56]. Need süsteemid võimaldavad patsientidel broneerida visiidiaegu, vaadata uuringute tulemusi ja retsepte ning teostada muid tervisega seotud toiminguid digitaalses keskkonnas. Isiklikud terviseandmed on nüüd lihtsamini kättesaadavad, muutes tervishoiuteenused mugavamaks nii patsientidele kui ka tervishoiutöötajatele. Siiski jäi andmete omandiõigus tervishoiuasutustele – nüüd digitaalsel kujul [56].

Isiklike terviseandmete digitaliseerimine tõi kaasa uusi väljakutseid. Üheks neist on selge arusaam terviseandmete väärtusest ja nende kasutamisest tervishoiuvaldkonna edasiseks arendamiseks. Kuidas neid andmeid kasutusele võtta, jääb endiselt lahtiseks küsimuseks.

Tervishoiusüsteemide edasiareng ja tulevik

Kolmas etapp on personaliseeritud lähenemine tervishoius. Suund on muuta tervishoiuteenused patsiendikeskseks, pakkudes rohkem funktsionaalsust ja individuaalset lähenemist. Personaliseeritud lähenemine arvestab ka igapäevaste tervisenäitajatega, võimaldades luua täpsema ettekujutuse konkreetse inimese tervisest ja elustiilist. Lisaks aitab see varakult parandada kahjulikke harjumusi ja õigeaegselt määrata raviplaani. Eestis pandi personaalmeditsiini rakendamisele alus 1999. aastal genoomi projekti ideega, mille raames on kogutud üle 200 000 geeniproovi, moodustades peaaegu 20% täisealisest elanikkonnast [57]. Alates aastast 2014 oli käivitatud personaalmeditsiini pilootprojekt [57].

Personaliseeritud tervishoiuteenuse teemat on käsitletud Tallinna Tehnikaülikooli ja Nortali koostöös kirjutatud töös [58]. Selle töö autorid annavad põhjaliku ülevaate praktilisest tegevuskavast, mille abil tervishoiuasutused saavad pakkuda personaliseeritud tervishoiuteenust. Lisaks, töö kirjeldab, kuidas kiirendatud meditsiinivaldkonna digitaliseerimise käigus lahendada nii kiireloomulisi väljakutseid, kui ka tulevasi nõudmisi. Põhiline tehniline lahendus, mis puudutab andmetega opereerimist, jääb aga samaks, nagu juba eksisteerivad süsteemid. Teiste sõnadega – andmete hoidmiseks ja vahendamiseks on mõeldud keskne ehk tsentraliseeritud süsteem. Selline lähenemine ei lahenda andmete hoidmise ja kasutamise dilemmasid, eriti andmete kuuluvuse ja omandiõiguse probleemi.

Käesolev lõputöö on samm järgmise ehk neljanda etapi suunas. Tegemist ei ole olemasolevast süsteemist loobumise ega selle ümbertegemisega, vaid loogilise järgmise arenguetapiga. Teine etapp lahendas andmetega opereerimise ebamugavuse ja tervishoiuteenuste kättesaadavuse probleeme. Kolmas etapp pakkus personaliseeritud tervishoiuteenuseid igale inimesele, aidates tõsta inimeste tervise ja elukvaliteeti. Neljas etapp hõlmab kõiki eelnevaid parendusi ja uuendusi ning lisaks lahendab ka varem lahendamata jäänud probleemid ja

väljakutsed.

4.4 Järeldused

Käesoleva lõputöö raames püstitatud eesmärk oli edukalt saavutatud, ning saadud tulemuste põhjal koostas autor nimekirja järeldustest, mis võtavad kokku tehtud töö tulemusi ning põhjendavad töö raames tehtud valikuid.

Käesoleva lõputöö raames süsteem oli arendatud vastavalt DSR (*Design Science Research*) põhimõttetele. Pakutud lahendus on osa suurest projektist (vt 4.1), mis pakub uuenduslikke lahendusi reaalsete tänasel päeval aktuaalsete probleemide lahendamiseks.

Järeldus 1: MAUI raamistik sobib tehnilise lahenduse teostamiseks

MAUI raamistiku valik osutus käesoleva töö teostamisel põhjendatuks mitmel olulisel põhjusel. Esiteks on C# programmeerimiskeele kompetents Eestis kõrgel tasemel, mis tähendab, et tulevikus on võimalik leida kvalifitseeritud arendajaid, kes suudavad süsteemi edasi arendada ja hooldada. Teiseks pakub C# rikkalikku valikut teeke ja tööriistu, mis lihtsustavad FHIR formaadis andmete töötlemist ja haldamist. Kolmandaks võimaldab MAUI raamistik arendada rakendust ühise koodibaasi põhjal nii mobiil- (iOS ja Android) kui ka lauaarvuti platvormidele (nt Windows ja macOS). See platvormide ülene lähenemine vähendab arenduskulusid, lihtsustab hooldust ja võimaldab kiiremat juurutamist mitmesugustes keskkondades.

Järeldus 2: Veebirakendus ei sobi ettenähtud lahenduse teostamiseks

Käesoleva töö raames olid vaadeldud peamised isiklike terviseandmetega seotud probleemid kolme dilemmana - isiklike terviseandmete ligipääsetavus, isiklike terviseandmete kogumine ja hoidmine, ning isiklike terviseandmete omandiõigus. Veebipõhise rakenduse puhul ei ole võimalik neid probleeme korraga lahendada. Veebipõhine lahendus eeldab üldjuhul andmete salvestamist tsentraalsesse serverisse, mis on sageli hallatud teenusepakkuja või organisatsiooni poolt. Selline arhitektuuriline ülesehitus tähendab, et andmed ei paikne patsiendi otsese kontrolli all, vaid sõltuvad kolmanda osapoole hallatavast infrastruktuurist. Lisaks kaasnevad veebilahendustega mitmed turvariskid, nagu autentimisega seotud nõrkused, andmelekked ning andmetele loata juurdepääs, mis võivad kahjustada patsiendi privaatsust ja autonoomiat.

Ühest küljest, veebipõhist lahendust võiks kasutada kasutaja ja detsentraliseeritud sisuadresseeritava salvestusvõrgu interaktsiooni vahelise kihina, mille rolli täidab käesoleva

lõputöö raames arendatud MAUI rakendus. Kuid sellisel juhul ei pruugi olla võimalik tagada rakenduse kasutamist ükskõik millise seadme peal ja selleks on mitu põhjust:

1. Veebilehitseja erinevused. Erinevad veebilehitsejad (nt Google Chrome, Mozilla Firefox, Safari, Microsoft Edge jt) tõlgendavad ja renderdavad HTML-, CSS- ja JavaScript-koodi teatud juhtudel erinevalt, mis võib kaasa tuua kasutajaliidese visuaalse või funktsionaalse ebajärjepidevuse. See omakorda vähendab rakenduse töökindlust ja usaldusväärsust, eriti olukordades, kus on oluline täpne ja ühtne andmete esitlus ning kasutajaliidese funktsionaalsus [59].
2. Veebilehitseja puudumine. Näiteks, jooksvalt terviseandmete kogumiseks oleks võimalik tagada süsteemile ligipääsu ka nutikellade kaudu. Aga nutikelladel ja muudel piiratud kasutajaliidesega seadmetel puudub täielikult veebilehitseja tugi või on see väga piiratud. Seega ei ole veebirakendus nendel platvormidel üldse kasutatav.

Järeldus 3: Terviseandmete salvestamiseks ja hoidmiseks sobib FHIR standard

FHIR andmete formaadi eesmärk on parandada terviseandmete vahetatavust ja ühilduvust erinevate süsteemide vahel [2].

FHIR põhineb hästi defineeritud andmemudelil, mis koosneb nn „ressurssidest“ (*resources*) – väikestest, taaskasutatavatest andmeüksustest, mis kirjeldavad tervishoius olulisi objekte, nagu diagnoosid, protseduurid, ravimid jms. See modulaarne ülesehitus võimaldab luua paindlikke, vajaduspõhiseid lahendusi ning samal ajal säilitada ühtse struktuuri. See oluliselt lihtsustab andmete standardiseerimist ja interpreteerimist [2].

Lisaks on FHIR laialt kasutusel ja toetatud erinevate tervishoiusüsteemide ja -tarnijate poolt üle maailma, mis suurendab selle usaldusväärsust ja jätkusuutlikkust. See tähendab, et FHIR-põhised lahendused on tulevikukindlad ning ühendatavad teiste FHIR-toega süsteemidega [2].

Järeldus 4: DICOM piltide salvestamine ja kuvamine on tehniliselt teostatav

Käesoleva lõputöö raames oli vajalik tõestada, et DICOM pildiandmed on võimalik salvestada detsentraliseeritud sisu-adresseeritavas salvestusvõrgus ning pärast ka küsida need tagasi ja kuvada sellisel kujul, mis esialgselt oli. DICOM pildiandmete üleslaadimine MAUI rakenduse kaudu ning salvestamine detsentraliseeritud sisu-adresseeritavas salvestusvõrgus toimib ootuspäraselt. DICOM pildiandmete pärimine detsentraliseeritud sisu-adresseeritavast süsteemist ning taastamine kuvamiseks MAUI rakenduse kaudu

samamoodi toimib ootuspäraselt. DICOM pildiandmetega opereerimisega seotud protsessi detailne kirjeldus on tehtud alapeatükis 3.5.

Järeldus 5: Detsentraliseeritud sisu-adresseeritavas salvestusvõrgus terviseandmete hoidmise ja kasutamise tehniline teostatavus

Isiklike terviseandmete hoidmine ja kasutamine detsentraliseeritud sisu-adresseeritavas salvestusvõrgus ja selle tehniline teostatavuse valideerimine oli käesoleva lõputöö raames püstitatud eesmärgiks. Käesoleva lõputöö raames arendatud süsteemi abil oli tõestatud, et tehniliselt pakutud lahendus on teostatav. Süsteemi põhimõtted, protsesside ja arhitektuuri kirjeldused on detailselt välja toodud peatükis 3. Tulemuste peatükis selgitatakse nii süsteemi kasutamise printsiibid, kui ka süsteemi tehniline teostatavus ja ülesehitus, k.a. süsteemi erinevate osade kirjeldamine ja nende osade vahelise interaktsiooni selgitamine.

Järeldus 6: Tervishoiuteenuse osutaja ligipääs terviseandmete omaniku andmetele

Praegusel hetkel on mitmeid lahendamata küsimusi, mis puudutavad tervishoiuteenuse osutaja ligipääsu terviseandmete omaniku andmetele detsentraliseeritud sisu-adresseeritavas salvestusvõrgus. Ühe võimalusena võiks terviseandmete omanik anda tervishoiuteenuse osutajale visiidi ajaks ajutise juurdepääsu, jagades peamist võtit krüpteeritud kujul. Selle meetodi puhul peaks aga tagama ka seda, et ligipääs terviseandmete omaniku andmetele kaoks kohe pärast visiidi lõppu ära, et terviseandmete privaatsust mitte rikkuda.

Teisest küljest on oluline arvestada tervishoiuteenuse osutaja tööprotsessi ja vajadust tutvuda patsiendi andmetega juba enne visiiti. See võib eeldada mehhanismi, mis võimaldab tervishoiuteenuse osutajal ajutiselt ligipääsu saada ka enne visiidi toimumist. Detsentraliseeritud sisu-adresseeritavas salvestusvõrgus terviseandmete sisu ei hakka sisaldama patsiendi isiksusele viitavat infot. Seega teoreetiliselt on võimalik, et tervishoiuteenuse osutaja omab tema patsientide võtmete nimekirja, mille alusel ta saab andmetele ligipääsu taotleda. Selline süsteem nõuab täiendavat juurdepääsukontrolli, et tagada andmete terviklikkust ning vältida volitamata ligipääsu.

Pakutud lahendus on autori hüpotees ning käesolevas töös ei käsitleta tervishoiuteenuse osutaja ligipääsu mehhanismide rakendamist. Edasised uuringud on vajalikud, et leida tasakaal terviseandmete omaniku ehk patsiendi privaatsuse ja tervishoiuteenuse osutaja töö efektiivsuse vahel detsentraliseeritud sisu-adresseeritava salvestusvõrgus andmete hoidmise kontekstis.

Järeldus 7: Seos EHDS eesmärkide ja printsiipidega

Arendades tervishoiusüsteeme ja pakkudes uusi lahendusi tuleb arvestada EHDS eesmärkide ja printsiipidega [17]. EHDS raamistik on ehitatud arvestades ka GDPR

regulatsioone [16]. See tähendab, et andmetega opereerimise reeglid ja printsiibid peavad olema rangelt täidetud. Järgnevates punktides kirjeldatakse EHDS eesmärgid ja kuidas käesoleva töö tulemused toetavad neid:

- EHDS eesmärk 1: Pakkuda inimestele digitaalset juurdepääsu ja kontrolli oma isiklike terviseandmete (PHR) üle [60].

Pakutud idee lahendab isiklike terviseandmete omandiõiguse probleemi ning tagab täieliku kontrolli oma andmete üle terviseandmete omaniku käes.

- EHDS eesmärk 2: Korraldada ühist ruumi elektrooniliste terviseandmete süsteemide, asjakohaste meditsiiniseadmete ja kõrge riskiga tehisintellekti süsteemide jaoks [60].

Kasutades detsentraliseeritud sisu-adresseeritava salvestusvõrku isiklike terviseandmete hoidmiseks pakutud lahendus võimaldab ühise andmete hoidmise ruumi loomist. Samal ajal, sellel ühisel ruumil ei ole tsentraliseeritud süsteemi puuduseid.

- EHDS eesmärk 3: Tagada usaldusväärne ja tõhus raamistik terviseandmete kasutamiseks teadusuuringutes, innovatsioonis ja ühiskonnas [60].

Lahendades isiklike terviseandmete kättesaadavuse, omandiõiguse ja anonümiseerimise probleemi, pakutud lahendus tagab nende andmete kasutamise võimalust teaduses ja innovatsioonis.

Võib järeldada, et käesoleva töö raames pakutud lahendus toetab PHR suunda ja EHDS eesmärke ning järelikult saab olla tervishoiuvaldkonna digitaalse arengu järgmiseks sammuks.

4.5 Praktilised õppetunnid tööprotsessist

Käesoleva lõputöö uurimis- ja arendusprotsessi käigus ilmnes aspekte, mida alguses ei osanud täielikult ette näha:

1. MAUI raamistiku abil ehitatud rakenduse ülesehituse ja konfigureerimise keerukused. Töö käigus tuli välja, et MAUI rakenduse seadistamine mitmemodulaarseks rakenduseks (vaadete ja andmete representeerimise moodul, andmete töötlemise ehk põhimoodul, testide moodul) vajab rohkem MAUI raamistiku spetsiifilisi teadmisi ja kogemust ning ei toimu autori poolt varem kasutatud raamistikkega ühtemoodi;
2. Swarm dokumentatsiooni puudused. Töö alguses tundus Swarm süsteemi doku-

mentatsioon korralik ja täielik, kaasa arvatud Bee API dokumentatsioon. Kuid töö käigus ilmnes, et Bee API vastu tehtavate päringute vastused ja eriti veavastused ei ole piisavalt kirjeldatud ja selgitatud. Selle tõttu kuulus autoril rohkem aega kui oli eeldatud selleks, et probleemidest aru saada *brute force*[1] stiilis uurimise käigus;

3. FHIR Shorthand formaadiks ja vastupidi andmete konverteerimine. FHIR Shorthand formaat näeb väga lihtne välja. Seetõttu töö alguses ei osanud ette näha, et kõige lihtsamaks formaadiks ja vastupidi konverteerimine võib tehniliselt osutuda kõige keeruliseks andmete konverteerimise ülesandeks. Keerukust toob sisse see aspekt, et FHIR Shorthand formaadi käsitlemiseks on olemas spetsiaalsed teegid, mis on kirjutatud JavaScript ja TypeScript programmeerimiskeeltel. See tähendab, et C# programmeerimiskeelel kirjutatud rakenduses neid teeke ei ole võimalik otseselt kasutada ning tuleb mõelda eraldiseiseva teenuse realiseerimise peale, või hoopis luua täiesti uut teeki C# programmeerimiskeelel.

Kui autor saaks töö algusest peale uuesti alustada, teeks mõningaid asju teisiti. Esiteks keskenduks autor projekti alguses rohkem süsteemi tehnilise implementatsioonile, mitte meditsiiniinformaatika valdkonnaga tutvumisele. Teiseks, arvestades käesoleva töö eesmärke ja oodatud tulemusi, keskenduks autor kohe projekti alguses rohkem just rakenduse ja Swarm süsteemi interaktsioonile ning terviseandmete töötlemise põhiloogikale, mitte rakenduse kasutajaliidese kujunduse valimisele. Kolmandaks loeks autor kõigepealt kas kogu "Book of Swarm" raamatu läbi või vähemalt potentsiaalselt vajalikke peatükke enne rakenduse arendamise algust.

Töö käigus sai autor väärtuslikke kogemusi ja teadmisi. Autor õppis:

1. kui oluline on teha detailse uurimistöö isegi iga väikese tehnilise valiku tegemisel, isegi kui tundub, et sarnane kogemus on juba olemas ning ülesanne tundub tuttav ja arusaadav;
2. kui oluline on kahtluse alla seada iga otsus, isegi see, mis näib kõige õigem antud ajahetkel;
3. et peab arvestama suurema ajakuluga ka nende planeeritud asjadele, mille kohta tundub, et kõik vajalik informatsioon on juba käes olemas;
4. ei tohi tähtaegasid karta ning alustada projekti tehnilise implementeerimise arendamisega enne, kui kõik isegi väiksemad detailid on selged.

Antud alapeatükis kirjeldatud kogemused on rakendatavad ka edaspidistes projekti sammudes ja autori enda teadustöös.

[1] *Brute force* - (eesti keeles sageli "toore jõu meetod") on primitiivne katse-eksituse meetod probleemi lahendamiseks, mille käigus proovitakse kõik võimalikud variandid järjest läbi, kuni leitakse õige lahendus.

4.6 Tuleviku uurimissuunad

Käesolev töö rajab alguse antud teema edasi arendamiseks nii tehnoloogilises, kui ka ühiskondlikus suunas. Autor toob välja praegusel hetkel teada olevad võimalikud jätkusuunad ning järgmised tegevus- ja uurimissammud.

1. **Patsiendi andmetele ligipääs arsti rollis:** kuidas jagada peamist võtit? Kas on vaja anda arstile ligipääsu andmetele juba enne visiiti?
2. **Teiste seadmete kasutamine enda andmete vaatamiseks:** kuidas oleks patsiendi jaoks võimalik vabalt kasutada ka teisi seadmeid, milles ei ole tema peamine võti salvestatud? Kas see teema kuulub võtme jagamise teema alla, nagu näiteks kirjeldatud punktis 1?
3. **Pereliikmete andmetele ligipääs:** kuidas tagada turvalisust pereliikmete (lapsed, vanemad) andmetele ligipääsule? Kuidas peamise võtme jagamine peaks toimima? Peamise võtme jagamise teema on tihedalt seotud olukorraga kirjeldatud punktis 1, seega võibolla saab olla käsitletud koos.
4. **Seadme kaotamine koos peamise võtmega:** kuidas võimaldada peamise võtme taastamist turvaliselt ilma patsiendi andmetele ligipääsu avalikustamist? Kas see teema saab olla samamoodi ühendatud ka punktidega 1 ja 2?
5. **Teised variandid peamise võtme hoidmiseks:** kas oleks võimalik turvaliselt hoida põhivõtit mitte kasutaja seadmes? Kas see võimaldaks veel vabamalt kasutada erinevad seadmed süsteemi ligipääsu saamiseks?
6. **Andmete sisestamine rakendusse kõnetöötluse abil:** oleks võimalik katsetada stsenaariumit, kui arst lihtsalt räägib informatsiooni patsiendi seisundist, ning rakendus oskab töödelda kõnet, transformeerida saadud info õigeteks FHIR ressurssideks ja tulemusena salvestada need Swarm võrgustikku.
7. **FHIR ressursside kategoriseerimise parendamine:** kas oleks võimalik jaotada FHIR ressurssid kasutajate jaoks veel mugavamateks ja intuitiivsemateks kategooriateks?
8. **BZZ tokenite arvu täiendamine:** kuidas seda võimaldada kasutajale mugavalt, automaatselt? Kas tohib anda rakendusel otsustada, millal juurde osta? Kuidas teavitada kasutajat, et DAI mündimärgid on otsas?

5. Kokkuvõte

Lõputöös käsitleti isiklike terviseandmetega opereerimisega probleeme, mida tänapäeval eksisteerivad süsteemid ei ole võimelised lahendama. Selleks on ligipääs andmetele, andmete terviklikkus, andmete omandamise õigused, andmete kasutamise võimalused andmete väärtuslikkuse kasutamine täies mahus.

Lõputöö eesmärk oli analüüsida, kavandada kasutusjuhud ja arendada API-d selleks, et valideerida detsentraliseeritud sisu-adresseeritavas salvestusvõrgus terviseandmete hoidmise ja kasutamise tehnilist teostatavust ja realiseerida TRL 3 tasemel [7] prototüüp selleks, et saaks hakata valideerima tehnoloogia kasutatavust kõrgematel TRL tasemetel.

Lõputöö tulemuste saavutamiseks esimese sammuna oli teostatud põhjalik tutvumine valdkonnaga. Sellele järgnes ärianalüüs, et tuvastada probleeme, mida tuleb lahendada ning uurida, miks tänapäeval meditsiinivaldkonnas eksisteerivad ning kasutuses olevad süsteemid ja tarkvaralised lahendused ei saa tuvastatud probleeme lahendada (või isegi tekitavad neid). Edasi hakkas tehniline osa tööst, mis sisaldas ideena pakutud süsteemi tehnilise analüüsi ja rakenduse arendust.

Lõputöö tulemina valmis pakutud süsteemi esialgne analüüs ning detsentraliseeritud sisu-adresseeritavas salvestusvõrgus terviseandmete hoidmise ja kasutamise tehnilise teostatavuse tõestamiseks sai arendatud prototüüp-rakendus tarkvara valmidustasemel TRL 3. Samuti olid läbi mõeldud tuleviku väljakutsed, uuritud detsentraliseeritud sisu-adresseeritavas salvestusvõrgu võimekused ja tehnilise implementeerimise võimalused.

Tulemuste põhjal järeldas autor, et uudse lahenduse kasutatavus ja võimekus on tõestatud vastavalt rakenduse valmidusastmele TRL 3.

Lõputööks püstitatud eesmärk oli saavutatud. Autor rõhutab, et käesolev töö on alles esimene, kuid juba piisavalt suur samm tervishoiuvaldkonna tehnoloogiate arengu järgmise etapi suunas. Käesoleva lõputöö järgmise sammuna plaanib autor jätkata oma uurimistööd doktorantuuris.

Kasutatud kirjandus

[1] Wikipedia. *Proof of concept*. https://en.wikipedia.org/wiki/Proof_of_concept. Online; Accessed: 06.01.2025. 2024.

[2] Wikipedia. *Fast Healthcare Interoperability Resources*. https://en.wikipedia.org/wiki/Fast_Healthcare_Interoperability_Resources. Online; Accessed: 24.04.2025.

[3] Wikipedia. *OSI model*. https://en.wikipedia.org/wiki/OSI_model. Online; Accessed: 18.03.2025.

[4] Wikipedia. *Health Level 7*. https://en.wikipedia.org/wiki/Health_Level_7. Online; Accessed: 24.04.2025.

[5] Ethereum Swarm. *BZZ Token*. https://www.ethswarm.org/get-bzz. Online; Accessed: 24.04.2025.

[6] Wikipedia. *Dai (cryptocurrency)*. https://en.wikipedia.org/wiki/Dai_(cryptocurrency). Online; Accessed: 24.04.2025.

[7] Wikipedia. *Technology readiness level*. https://en.wikipedia.org/wiki/Technology_readiness_level. Online; Accessed: 06.01.2025. 2024.

[8] DICOM Standard. *About DICOM: Overview*. https://www.dicomstandard.org/about. Online; Accessed: 28.12.2024.

[9] Wikipedia. *Design science (methodology)*. https://en.wikipedia.org/wiki/Design_science_(methodology). Online; Accessed: 18.03.2025.

[10] Wikipedia. *Web3*. https://en.wikipedia.org/wiki/Web3. Online; Accessed: 06.03.2025.

[11] Mare Ruuge and Marika Inno. *Tervishoiukulud 2020*. Tervise Arengu Instituut. 2022.

[12] Steve Alder. *Healthcare Data Breach Statistics*. https://www.hipaajournal.com/healthcare-data-breach-statistics/. Online; Accessed: 18.02.2025.

[13] Swarm. *What is Swarm?* https://docs.ethswarm.org/docs/learn/technology/what-is-swarm. Online; Accessed: 26.09.2024. 2024.

[14] Toomas Klementi, Gunnar Piho, and Peeter Ross. “A reference architecture for personal health data spaces using decentralized content-addressable storage networks”. In: *Frontiers in Medicine* 11 (2024), p. 1411013.

[15] *Ettevõtte andmebaasist laeti ebaseaduslikult alla Apotheka kliendikaartide omanike andmeid*. https://www.aki.ee/uudised/ettevotte-andmebaasist-laeti-ebaseaduslikult-alla-apotheka-kliendikaartide-omanike-andmeid. Andmekaitse Inspektsioon. Online; Accessed: 09.01.2025. 2024.

[16] Ben Wolford. *What is GDPR, the EU's new data protection law?* https://gdpr.eu/what-is-gdpr/. Online; Accessed: 06.01.2025.

[17] Cyber Risk GmbH. *The European Health Data Space*. https://www.european-health-data-space.com/. Online; Accessed: 06.01.2025.

[18] James N. Gray. “An approach to decentralized computer systems”. In: *IEEE Transactions on Software Engineering* SE-12.6 (1986), pp. 684–692. DOI: 10.1109/TSE.1986.6312966.

[19] Microsoft. *What is .NET MAUI?* https://learn.microsoft.com/en-us/dotnet/maui/what-is-maui?view=net-maui-9.0. Online; Accessed: 06.01.2025.

[20] Etherium. *Swarm*. https://github.com/ethersphere. Online; Accessed: 01.09.2024.

[21] Etherium. *What is Etherium*. https://ethereum.org/en/what-is-ethereum/. Online; Accessed: 01.09.2024.

[22] Swarm. *Working with Bee*. https://docs.ethswarm.org/docs/bee/working-with-bee/introduction. Online; Accessed: 06.01.2025.

[23] IPFS. *What is IPFS*. https://docs.ipfs.tech/concepts/what-is-ipfs/. Online; Accessed: 06.01.2025.

[24] DDB. *Mis on persoona ja miks see on oluline?* https://ddb.ee/mis-on-persoona-ja-miks-see-on-oluline/. Online; Accessed: 05.01.2025.

[25] Swarm. *Chunk Types - Single Owner Chunks*. https://docs.ethswarm.org/docs/develop/tools-and-features/chunk-types/#single-owner-chunks. Online; Accessed: 06.04.2025.

[26] Viktor Trón. *The Book of Swarm*. Vol. 2.2.3. 2024. ISBN: 978-615-01-9983-2. URL: https://www.ethswarm.org/The-Book-of-Swarm-2.pdf.

[27] Lilia Tünts. *GitLab - MedicalDataApp - SOC branch*. https://gitlab.cs.taltech.ee/litunt/medical-data-app/-/tree/med-24-single-owner-chunk-for-main-hash. Online; Accessed: 06.05.2025.

[28] Bee API. *Upload file or a collection of files*. https://docs.ethswarm.org/api/#tag/BZZ/paths/~1bzz/post. Online; Accessed: 09.11.2024.

[29] Bee API. *Get file or index document from a collection of files*. https://docs.ethswarm.org/api/#tag/BZZ/paths/~1bzz/get. Online; Accessed: 09.11.2024.

[30] Bee API. *Upload chunk*. https://docs.ethswarm.org/api/#tag/Chunk/paths/~1chunks/post. Online; Accessed: 09.11.2024.

[31] Bee API. *Upload data*. https://docs.ethswarm.org/api/#tag/Bytes/paths/~1bytes/post. Online; Accessed: 09.11.2024.

[32] DICOM Standards Committee. *DICOM PS3.1 2025a - Introduction and Overview*. https://dicom.nema.org/medical/dicom/current/output/html/part01.html. Online; Accessed: 06.04.2025.

[33] O.S. Pianykh. *Digital imaging and communications in medicine (DICOM): A practical introduction and survival guide*. Springer, Jan. 2008, pp. 1–383. ISBN: 9783540745709. DOI: 10.1007/978-3-540-74571-6.

[34] DICOM Viewer. *DICOM Viewer*. https://dicomviewer.net/viewer. Online; Accessed: 28.12.2024.

[35] Bee API. *Get referenced data*. https://docs.ethswarm.org/api/#tag/Bytes/paths/~1bytes/get. Online; Accessed: 09.11.2024.

[36] HL7 FHIR. "Resource Guide – FHIR v5.0.0". In: *Guide to Resources* Release 5 ().

[37] TEHIK. *Andmevaatur*. https://teabekeskus.tehik.ee/et/teenused/tis-teenused/portaalid/andmevaatur. Online; Accessed: 31.03.2025.

[38] TEHIK. *Andmevaaturi kasutusjuhend*. https://koodivaramu.eesti.ee/tehik/teabekeskus/dokumentatsioon/-/blob/master/teenused/Andmevaatur/Andmevaaturi%20kasutusjuhend.pdf?ref_type=heads. Online; Accessed: 31.03.2025.

[39] Swarm. *Postage Stamps*. https://docs.ethswarm.org/docs/concepts/incentives/postage-stamps. Online; Accessed: 06.05.2025.

[40] Refactoring Guru. *Factory Method*. https://refactoring.guru/design-patterns/factory-method. Online; Accessed: 10.05.2025.

[41] Lilia Tünts. *GitLab - MedicalDataApp - RDF branch*. https://gitlab.cs.taltech.ee/litunt/medical-data-app/-/tree/med-23-rdf-parser. Online; Accessed: 06.05.2025.

[42] GitHub FHIR. *FHIR/SUSHI*. https://github.com/FHIR/sushi. Online; Accessed: 06.05.2025.

[43] GitHub FHIR. *FHIR/GoFSH*. https://github.com/FHIR/GoFSH. Online; Accessed: 06.05.2025.

[44] FSH Online. *FSH Online*. https://fshonline.fshschool.org/. Online; Accessed: 06.05.2025.

[45] Wikipedia. *Lazy Loading*. https://en.wikipedia.org/wiki/Lazy_loading. Online; Accessed: 06.05.2025.

[46] medDream. *DICOM Library*. https://www.dicomlibrary.com/. Online; Accessed: 06.05.2025.

[47] Apache Jmeter. *What can I do with it?* https://jmeter.apache.org/index.html. Online; Accessed: 06.03.2025.

[48] ETIS. *Teadus- ja arendusprojekt - Terve Ühiskonna Digitervishoid (TEM-TA105)*. https://www.etis.ee/Portal/Projects/Display/8ad3d5f4-2390-4ce6-b37b-4e9cd819ea64. Online; Accessed: 06.04.2025.

[49] ETIS. *Teadus- ja arendusprojekt - Düslipideemia patsientide ravisoostumus ja ravi efektiivsus ning ravi tulemustele orienteeritud uudne patsiendi digitaalne tugirakendus (PRG2629)*. https://www.etis.ee/Portal/Projects/Display/575c079c-1ced-4a54-8cba-6ea64c042504. Online; Accessed: 06.04.2025.

[50] Fabio Duarte. *Amount of Data Created Daily*. https://explodingtopics.com/blog/data-generated-per-day. Online; Accessed: 10.12.2024. 2024.

[51] Kevin Bartley. *Big data statistics: How much data is there in the world?"* https://rivery.io/blog/big-data-statistics-how-much-data-is-there-in-the-world/. Online; Accessed: 10.12.2024. 2024.

[52] GrowthPro. *Big Data Trends: 2024 and Beyond*. https://bigdata.growth.pro/blog/big-data-trends-2024/. Online; Accessed: 10.12.2024. 2024.

[53] Brad Porter. *What To Do About Healthcare's 'Messy Desk' Data Dilemma*. https://www.forbes.com/councils/forbestechcouncil/2023/12/12/what-to-do-about-healthcares-messy-desk-data-dilemma/. Online; Accessed: 20.12.2024. 2023.

[54] *Transforming Healthcare: the Power and Promise of Big Data*. https://www.bigdataframework.org/knowledge/transforming-healthcare-the-power-and-promise-of-big-data/. Big Data Framework. Online; Accessed: 06.01.2025.

[55] Indrek Kald. *PERHi juhatuse liige: tervishoius kasvab andmete hulk meeletu kiirusega*. https://www.ituudised.ee/uudised/2020/11/11/perhi-juhatuse-liige-tervishoius-kasvab-andmete-hulk-meeletu-kiirusega. Online; Accessed: 06.01.2025.

[56] TEHIK. *Tervise infosüsteem*. https://www.tehik.ee/tervise-infosusteem. Online; Accessed: 06.04.2025.

[57] Helen Lepa. *Personaalmeditsiini areng Eestis*. https://www.tai.ee/et/personaalmeditsiini-uudiskirjad/personaalmeditsiini-areng-eestis. Online; Accessed: 06.04.2025.

[58] Kertti Merimaa, Grace Smith, Tanel Ross, and Andres Raieste. "Personal Healthcare: Building Healthcare Systems for the Future". In: *TalTech Data Repository* (2024). DOI: 10.48726/e5h9k-f8y71.

[59] Fatema El Asabi. *Why Cross-browser Compatibility Matters In Web Development*. https://sopriza.com/why-cross-browser-compatibility-matters-in-web-development/. Online; Accessed: 24.04.2025.

[60] European Commission. *What is the EHDS Regulation about*. https://health.ec.europa.eu/ehealth-digital-health-and-care/european-health-data-space-regulation-ehds_en. Online; Accessed: 06.01.2025.

Lisa 1 – Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks[1]

Mina, Lilia Tünts

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose "Isiklike terviseandmete detsentraliseeritud sisu-adresseeritavas salvestusvõrgus hoidmise ja kasutamise kontseptsiooni valideerimine", mille juhendaja on Gunnar Piho ja Toomas Klementi
   1.1. reprodutseerimiseks lõputöö säilitamise ja elektroonse avaldamise eesmärgil, sh Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
   1.2. üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. Olen teadlik, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autorile.
3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

12.05.2025

[1] Lihtlitsents ei kehti juurdepääsupiirangu kehtivuse ajal vastavalt üliõpilase taotlusele lõputööle juurdepääsupiirangu kehtestamiseks, mis on allkirjastatud teaduskonna dekaani poolt, välja arvatud ülikooli õigus lõputööd reprodutseerida üksnes säilitamise eesmärgil. Kui lõputöö on loonud kaks või enam isikut oma ühise loomingulise tegevusega ning lõputöö kaas- või ühisautor(id) ei ole andnud lõputööd kaitsvale üliõpilasele kindlaksmääratud tähtajaks nõusolekut lõputöö reprodutseerimiseks ja avalikustamiseks vastavalt lihtlitsentsi punktidele 1.1. ja 1.2, siis lihtlitsents nimetatud tähtaja jooksul ei kehti.

Lisa 2 - Jmeter raport

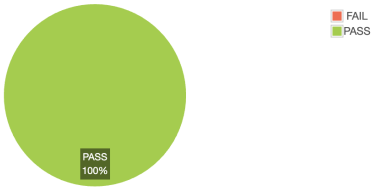
Test and Report information

Source file	"Result_test.jtl"
Start Time	"12/11/24, 12:43 AM"
End Time	"12/11/24, 12:43 AM"
Filter for display	""

APDEX (Application Performance Index)

Apdex	T (Toleration threshold)	F (Frustration threshold)	Label
0.824	**500 ms**	**1 sec 500 ms**	**Total**
0.824	500 ms	1 sec 500 ms	HTTP Request

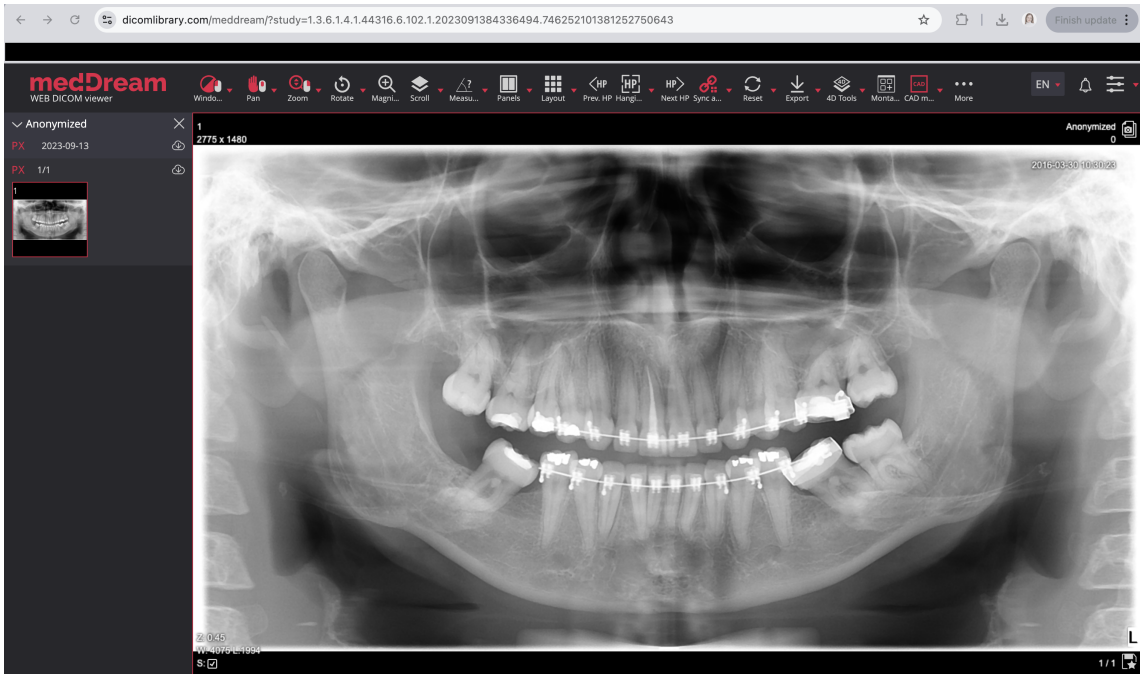
Requests Summary



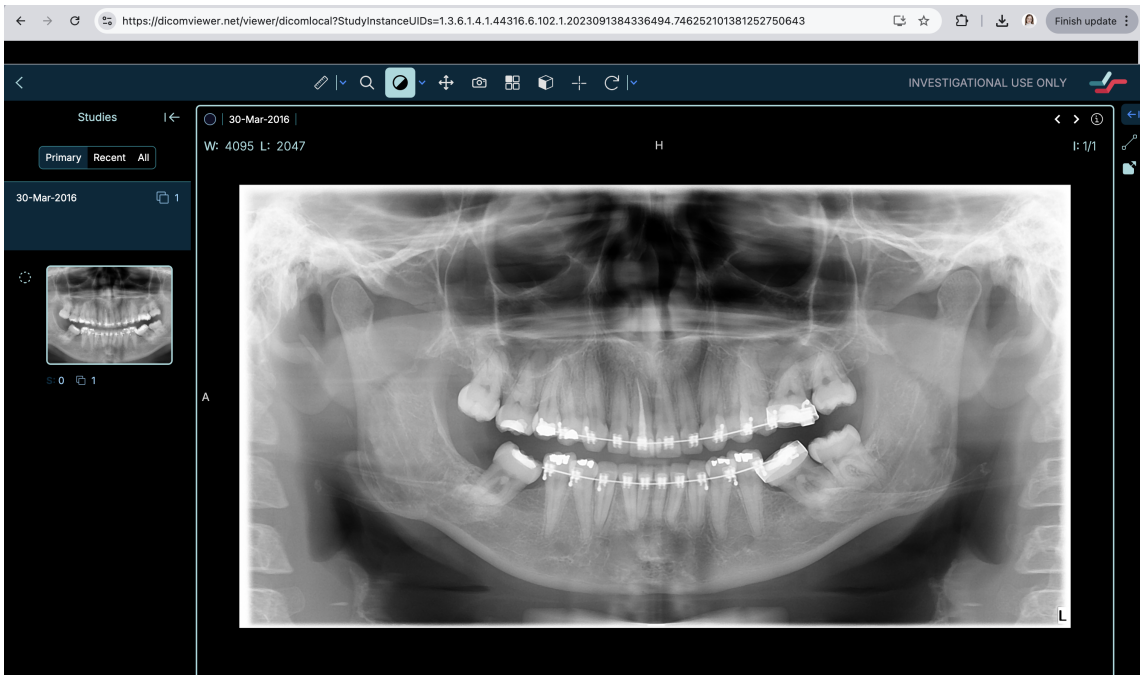
Statistics

Requests	Executions			Response Times (ms)							Throughput	Network (KB/sec)	
Label	#Samples	FAIL	Error %	Average	Min	Max	Median	90th pct	95th pct	99th pct	Transactions/s	Received	Sent
Total	**1000**	**0**	**0.00%**	**497.28**	**33**	**1153**	**216.00**	**1064.00**	**1103.00**	**1146.98**	**510.20**	**285.50**	**93.67**
HTTP Request	1000	0	0.00%	497.28	33	1153	216.00	1064.00	1103.00	1146.98	510.20	285.50	93.67

Lisa 3 - DICOM pilt testimiseks esialgsel kujul



Lisa 4 - DICOM pilt testimiseks taastatud DCAS võrgust



Lisa 5 - ImagingStudy FHIR ressurss DICOM faili viitega

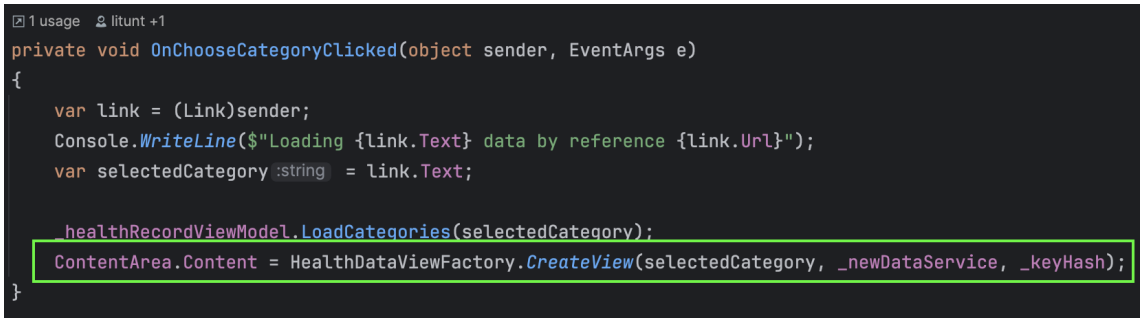
Lisa 6 - Observation FHIR ressurss, mis viitab ImagingStudy ressurssile

```
{
  "resourceType": "Observation",
  "status": "final",
  "category": [
    {
      "coding": [
        {
          "system": "http://terminology.hl7.org/CodeSystem/observation-category",
          "code": "imaging",
          "display": "Imaging"
        }
      ]
    }
  ],
  "code": {
    "coding": [
      {
        "system": "http://loinc.org",
        "code": "11503-0",
        "display": "Chest X-ray study"
      }
    ],
    "text": "Chest X-ray interpretation"
  },
  "subject": {
    "reference": "Patient/example",
    "display": "Jane Doe"
  },
  "effectiveDateTime": "2024-12-28T10:00:00Z",
  "performer": [
    {
      "reference": "Practitioner/example",
      "display": "Dr. Radiologist"
    }
  ],
  "valueString": "No acute abnormalities detected. Mild scoliosis observed.",
  "derivedFrom": [
    {
      "reference": "ImagingStudy/BBBBB…"

    }
  ]
}
```

Lisa 7 - DiagnosticReport FHIR ressurss, mis seob ImagingStudy ja Observation kokku

```
{
  "resourceType": "DiagnosticReport",
  "status": "final",
  "category": [
    {
      "coding": [
        {
          "system": "http://terminology.hl7.org/CodeSystem/v2-0074",
          "code": "RAD",
          "display": "Radiology"
        }
      ]
    }
  ],
  "code": {
    "coding": [
      {
        "system": "http://loinc.org",
        "code": "18748-4",
        "display": "Radiology report"
      }
    ]
  },
  "subject": {
    "reference": "Patient/example",
    "display": "Jane Doe"
  },
  "effectiveDateTime": "2024-12-28T10:00:00Z",
  "performer": [
    {
      "reference": "Practitioner/example",
      "display": "Dr. Radiologist"
    }
  ],
  "result": [
    {
      "reference": "Observation/CCCCC..."
    }
  ],
  "imagingStudy": [
    {
      "reference": "ImagingStudy/BBBBB..."
    }
  ]
}
```

Lisa 8 - Andmete sisestamise vaate loomine vastavalt valitud tüübile



```
1 usage  litunt +1
private void OnChooseCategoryClicked(object sender, EventArgs e)
{
    var link = (Link)sender;
    Console.WriteLine($"Loading {link.Text} data by reference {link.Url}");
    var selectedCategory :string  = link.Text;

    _healthRecordViewModel.LoadCategories(selectedCategory);
    ContentArea.Content = HealthDataViewFactory.CreateView(selectedCategory, _newDataService, _keyHash);
}
```

Lisa 9 - Terviseandmete sisestamise vaate loomise klass (HealthDataViewFactory)

```
1 usage  lilia
public static class HealthDataViewFactory
{

    1 usage  lilia
    public static HealthDataView CreateView(string category, NewFhirDataService newDataService, string keyHash)
    {
        return category switch
        {
            "Blood Pressure" => new HealthDataView(
                HealthRecordTypeEnum.BloodPressure,
                fhirType: "Observation",
                category: "Clinical Findings",
                newDataService,
                keyHash,
                params inputConfigs: ( name: "UpperBloodPressure",  placeholder: "Upper"),
                ( name: "LowerBloodPressure",  placeholder: "Lower")),

            "Heart Rate" => new HealthDataView(
                HealthRecordTypeEnum.HeartRate,
                fhirType: "Observation",
                category: "Clinical Findings",
                newDataService,
                keyHash,
                ( name: "HeartRate",  placeholder: "BpM")),

            "Steps" => new HealthDataView(
                HealthRecordTypeEnum.Steps,
                fhirType: "Observation",
                category: "Activity",
                newDataService,
                keyHash, ( name: "Steps",  placeholder: "Steps per day")),

            _ => throw new ArgumentOutOfRangeException(nameof(category), category, null)
        };
    }

}
```

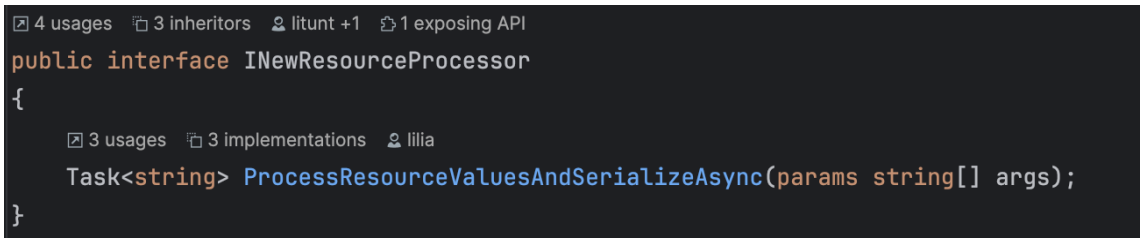
Lisa 10 - Sisestatud terviseandmete väärtuste kättesaamine, konverteerimine ja salvestamiseks edastamine

```
1 usage  lilia *
private async void OnSubmit(object sender, EventArgs e)
{
    var values = new List<string>();

    foreach (var editor in _inputFields.Values)
    {
        values.Add(editor.Text);
    }

    Console.WriteLine($"Values: {string.Join(", ", values)}");

    try
    {
        var jsonInput :string  = await DataProcessorFactory
            .GetDataProcessor(_recordType) // INewResourceProcessor
            .ProcessResourceValuesAndSerializeAsync(values.ToArray()); // Task<string>

        Console.WriteLine($"Updated JSON: {jsonInput}");

        try
        {
            await _newDataService.UploadFhirData(_fhirType, _category, jsonInput, _keyHash);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Error while updating FHIR data: {ex}");
        }
    }
    catch (Exception ex)
    {
        Console.WriteLine($"Error while parsing input JSON: {ex}");
    }

    foreach (var editor in _inputFields.Values)
    {
        editor.Text = "";
    }

    MessagingCenter.Send( sender: this,  message: $"Hide{_recordType}View");
}
```

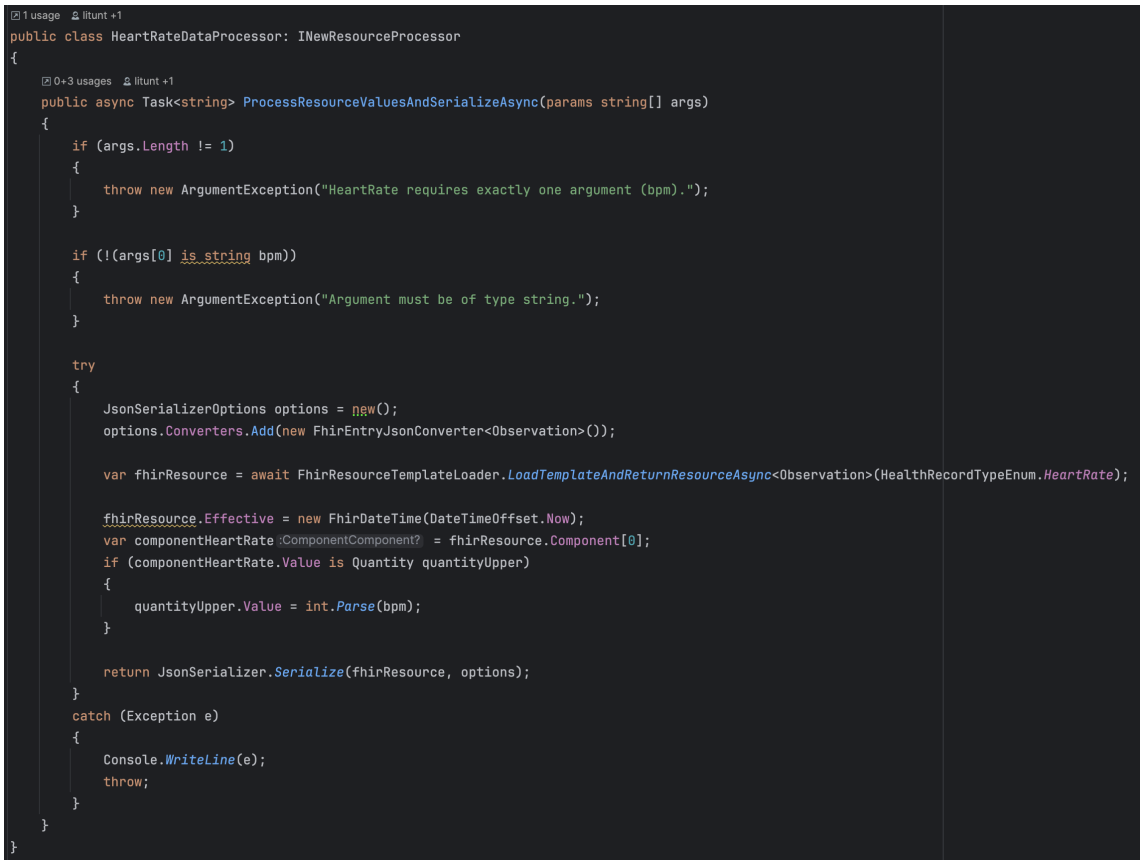
Lisa 11 - Andmete protsessori valimise abistav klass

```
3 usages  litunt
public static class DataProcessorFactory
{
    3 usages  litunt
    public static INewResourceProcessor GetDataProcessor(HealthRecordTypeEnum recordType)
    {
        switch (recordType)
        {
            case HealthRecordTypeEnum.BloodPressure:
                return new BloodPressureDataProcessor();
            case HealthRecordTypeEnum.HeartRate:
                return new HeartRateDataProcessor();
            case HealthRecordTypeEnum.Steps:
                return new StepsDataProcessor();
            default:
                throw new NotImplementedException($"No processor found for {recordType}");
        }
    }
}
```

Lisa 12 - Andmete ressurssi protsessori liides



Lisa 13 - Südamelöögi sageduse terviseandmete protsessor

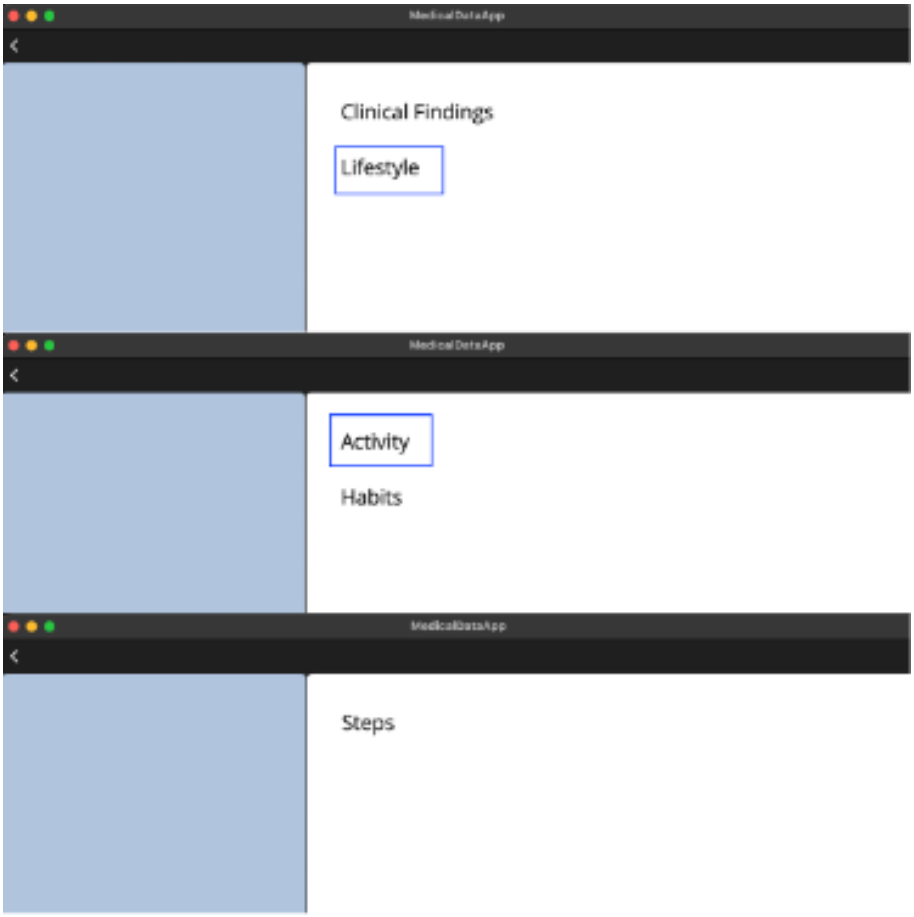


```
1 usage  liitunt +1
public class HeartRateDataProcessor: INewResourceProcessor
{
    0+3 usages  liitunt +1
    public async Task<string> ProcessResourceValuesAndSerializeAsync(params string[] args)
    {
        if (args.Length != 1)
        {
            throw new ArgumentException("HeartRate requires exactly one argument (bpm).");
        }

        if (!(args[0] is string bpm))
        {
            throw new ArgumentException("Argument must be of type string.");
        }

        try
        {
            JsonSerializerOptions options = new();
            options.Converters.Add(new FhirEntryJsonConverter<Observation>());

            var fhirResource = await FhirResourceTemplateLoader.LoadTemplateAndReturnResourceAsync<Observation>(HealthRecordTypeEnum.HeartRate);

            fhirResource.Effective = new FhirDateTime(DateTimeOffset.Now);
            var componentHeartRate :ComponentComponent? = fhirResource.Component[0];
            if (componentHeartRate.Value is Quantity quantityUpper)
            {
                quantityUpper.Value = int.Parse(bpm);
            }

            return JsonSerializer.Serialize(fhirResource, options);
        }
        catch (Exception e)
        {
            Console.WriteLine(e);
            throw;
        }
    }
}
```

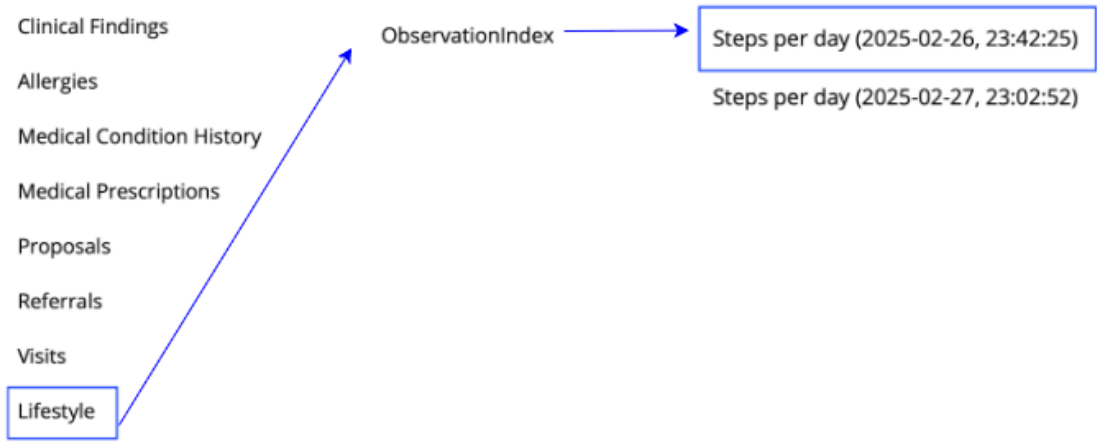
Lisa 14 - FHIR ressurssi JSON formaadi konverter

```
6 usages  litunt
public class FhirEntryJsonConverter<T>: JsonConverter<T> where T : Hl7.Fhir.Model.Resource
{
    litunt
    public override T Read(ref Utf8JsonReader reader, Type typeToConvert, JsonSerializerOptions options)
    {
        using (JsonDocument doc = JsonDocument.ParseValue(ref reader))
        {
            string entryJson = doc.RootElement.GetRawText();

            var fhirParser = new FhirJsonParser();
            return fhirParser.Parse<T>(entryJson);
        }
    }

    litunt
    public override void Write(Utf8JsonWriter writer, T value, JsonSerializerOptions options)
    {
        var fhirSerializer = new FhirJsonSerializer();
        string serializedEntry = fhirSerializer.SerializeToString(value);

        writer.WriteRawValue(serializedEntry);
    }
}
```

Lisa 15 - Terviseandmete kategooria ja alamkategooria valimine andmete sisestamiseks



Lisa 16 - Terviseandmete kategooria ja alamkategooria valimine andmete vaatamiseks



Lisa 17 - DICOM pildifaili salvestamine ja laadimine DCAS süsteemist