TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Keith Roland Lepik 185377IABB

# PIKTOGRAMME KASUTAVA PUUDEGA INIMESE JA TEGEVUSJUHENDAJA VAHELISE KOMMUNIKATSIOONI RAKENDUS

Bakalaureusetöö

Juhendaja: Liisa Jõgiste
MSc

Tallinn 2022

# 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.

Autor: Keith Roland Lepik

13.05.2022

# Annotatsioon

Lõputöö eesmärgiks oli luua rakendus, mille kasutamisega on võimaldatud puudega inimese ja tegevusjuhendaja vaheline suhtlus internetis kasutades piktogramme. Töö eesmärgi täitmiseks analüüsiti turul pakutavaid piktogrammidega suhtlust võimaldavaid rakendusi. Seejärel viidi läbi intervjuu hooldekodus igapäevaselt puudega inimeste tegevusi juhendavate töötajatega. Rakenduste analüüsi ja töötajatega vestluse alusel pandi kirja nõuded uuele rakendusele.

Nõuete alusel disainiti rakendusele arhitektuur ja arhitektuuri alusel loodi rakendus, millega on võimalik töötajal laadida üles pilte rakendusse ja määrata igale kliendile temale sobivad piktogrammid. Kliendil on võimalik temale määratud piktogramme saata tegevusjuhendajale ja tegevusjuhendajal on võimalik saabunud piktogramme näha.

Rakenduse arendamise jooksul järgiti tarkvaraarenduse tavasid, mis on kiidetud heaks mitme noteeritud tarkvaraarendaja poolt. Heade tavade kasutamine tagab rakenduse lihtsat skaleerimist ja lihtsutab oluliset muudatuste tegemist.

Lõputöö on kirjutatud eesti keeles ning sisaldab 31 leheküljel, 5 peatükki, 13 joonist, 1 tabelit.

# Abstract

## APPLICATION TO ALLOW THE COMMUNICATION BETWEEN A DISABLED PERSON AND AN ACTIVITY INSTRUCTOR USING PICTOGRAMS

The aim of this thesis was to create an application that enables a communication channel between a disabled person and an activity supervisor using pictograms. In order to reach the goal the author constructed an analysis of applications that enable the communication between users using pictograms. After that an interview was conducted with individuals who are currently employed in a care home. The outcome of that interview was consolidated into the requirements that were set on the application in development.

Given the set of requirements the author first designed an architectual view of the system and then proceeded with developing the application. The developed application allows organizational managers to upload pictograms into the developed system, add new clients, add new houses and assign pictograms to a client. Clients can then use the application to send their assigned images to the house for a one-way communication. The house can view recieved messages from clients and act based on the needs of the client.

During the development process of the system, the author followed software development principles, that are considered good by many great software engineers. This allows the application to be scalable and easily modifyable.

The thesis is in Estonian and contains 31 pages of text, 5 chapters, 13 figures, 1 table.

# Sisukord

# Jooniste loetelu

# Tabelite loetelu

# Lühendite ja mõistete loetelu

| | |
|---|---|
| API | *Application Programming Interface* – rakenduse programmeerimise liides |
| *Cache* | Tarkvara osa, milles hoitakse ajutisi andmeid |
| CLI | *Command-Line Interface* – käsureale loodud liides |
| DTO | *Data Transfer Object* – andmevahetuseks kasutatav objekt |
| IDE | *Integrated Development Environment* – integreeritud arenduskeskkond |
| IOS | Apple-i poolt loodud operatsioonisüsteem |
| UML | *Unified Modeling Language* – ühendatud mudelleerimise keel |
| Klient | Hooldekodus olev puudega isik |
| Maja | Hooldekodus ühes majas asuv tahvelarvuti |
| MartenDB | Dokumentidena struktureeritud andmebaasi skeemi loomise raamistik |
| MinIO | Failide haldamise süsteem |
| Moq | Testimisel kasutatav objektist simulatsiooni loomise raamistik |
| NUnit | Ühiktestide loomise raamistik |
| PostgreSQL | Relatsioonlise andmebaasi server |
| PWA | *Progressive Web Application* – progressiivne veebirakendus |
| Vue.Js | Komponendipõhist arendust võimaldav javascripti raamistik |

# 1 Sissejuhatus

Puudega inimese ja tegevusjuhendaja vaheline kommunikatsioon ei ole infoajastu kiire arenguga kaasa liikunud. Praktiliselt igaks elujuhtumiks on loodud mitmeid rakendusi, millega on võimalik suures koguses informatsiooni jagada millisekunditega. Kui selliseid rakendusi võib nimetada mitmeid, näiteks *Facebook Messenger*, *Discord* jpm, siis otsides rakendust, millega on võimalik raske puudega inimesel, kellel ei ole oskusi ja võimalusi kirjutada sidusaid tekste, äärmiselt raske leida. Turul pakutakse ka kõnet genereerivaid masinaid, mis on kallid ja nende soetamine haiglatesse või peredesse ei ole sageli majanduslikult võimalik.

Puudega inimesed, kes ei suuda kõnega või tekstina oma tundeid ja soove avaldada, kasutavad kommunikeerimiseks piktogramme. Piktogramm kujutab endast pilti, millel on kujutatud tegevust või tunnet [1]. Töö jooksul läbi viidud intervjuust selgub, et kommunikatsioon puudega inimeste ja tegevusjuhendajate vahel hoolekande kodudes toimub hetkel füüsiliselt välja prinditud piktogramme kasutades. Kasutusel olev lahendus on kohmakas, piiratud võimekusega ja nõuab tegevusjuhendaja füüsilist kohalolu.

Bakalaureusetöö eesmärgiks on analüüsida kasutusel oleva lahenduse kitsaskohti ja luua rakendus, mille abil on võimalik puudega inimesel oma tundeid ja erisoove kiiremini avaldada.

Käesoleva lõputöö tööprotsess jaguneb etappideks järgmiselt:

- hetkeolukorra uurimine,
- nõuete alusel arhitektuuri loomine,
- rakenduse arendamine ja
- Rakenduse demo esitlemine.

Töö esimeses etapis vaadatakse täpsemalt, mis on kommunikeerimisprotsess ja selle alusel otsitakse alternatiivseid tarkvarasid puudega inimestega suhtlemiseks ja seejärel pannakse kirja intervjuust välja tulnud nõuded rakendusele.

Töö teises etapis loob autor rakenduse arhitektuuri kondikava. Arhitektuuri loomisel lähtutakse eelnevalt välja toodud nõuetele.

Kolmandas etapis valitakse välja sobivad arenduskeskkonnad ja tarkvara raamistikud. Etapis kirjeldatakse ka ära arendusprotsess.

Neljandas etapis on välja toodud töö käigus loodud rakenduse kasutajaliidese pildid ja peamised tulemused. Rakendust demonstreeritakse ka hooldekodu töötajatele.

# 2 Hetkeolukord

## 2.1 Puuetega inimeste kommunikeerimisprotsess

Puude raskusastmeid võib jagada kolme suuremasse kategooriasse: sügav, raske ja keskmine puue. Sügava puudega isik vajab ööpäevaringset erihooldusteenust. Raske puudega isik vajab igapäevaselt erihooldusteenust. Keskmise puudega isik vajab erihooldusteenust iganädalaselt. [2]

Suhtlemine jaguneb kahte suuremasse kategooriasse: verbaalne ja mitteverbaalne. Mitteverbaalseks suhtluseks loetakse suhtlemist, mis ei toimu sõnadega. Selleks võib olla näiteks hääletoon, miimika, piktogrammid jne. [3]

Kommunikatsioon on suhtlemise protsess, mille käigus vahetatakse informatsiooni. Kommunikatsiooni põhielemendid on:

- allikas - sõnumi algataja,
- vastuvõtja - sõnumi saaja,
- vahend - kanal (teine inimene, piktogrammid, raamat),
- sõnum - teade, märk, sümbol,
- mõju - muutus, mis toimub vastuvõtjas,
- efektiivsus - kas sõnumi saatja kavatsus täitus,
- müra - informatsiooni edastamisel ja vastuvõtmisel vahepealsed segajad,
- tagasiside - reaktsioon sõnumile.

[3]

Antud bakalaureusetöös keskendutakse peamiselt vahendi ehk kanali parandamisele.

## 2.2 Alternatiivsed tarkvarad

Antud peatükis analüüsib autor nelja suuremat piktogrammidega kommunikeerimist võimaldavat rakendust. Kõikide alternatiivsete rakenduste puhul on peamine fookus

kõne genereerimisel. Kõne genereerivad rakendused on loodud peamiselt lokaalseks kasutamiseks, kus pildile vajutades kõlab helisignaal, mis kirjeldab pilti [4].

Turul pakutavatest rakendustest, mille eesmärk on piktogrammidega suhtlust võimaldada, valis autor välja järgnevad: Talk Up! Pictograms Communicator [5], JABtalk [6], SymboTalk - AAC Talker [7], PicCom [8]. Vaatluse all olevate tarkvarade analüüsi käigus kaardistatakse olemasolevate tarkvarade tugevused ja nõrkused.

Tabel 1 Alternatiivsed rakendused ja nende funktsionaalsused

| | **Talk Up!** | **JabTalk** | **SymboTalk** | **PicCom** |
|---|---|---|---|---|
| **Platvorm** | Android 9 | Android 4.0.3 | Android 4.4 | Android 4.1 |
| **Hind** | 0 | 0 | 0 | 0 |
| **Piktogrammidega lausete koostamine** | Jah | Jah | Jah | Jah |
| **Piktogrammide kuvamine helisignaalidena** | Jah | Jah | Jah | Jah |
| **Saab saata piktogrammidega koostatud lauseid hooldajale** | Ei | Ei | Ei | Ei |
| **Võimalus koostada oma piktogrammide laud** | Ei | Jah | Jah | Jah |
| **Saab luua oma piktogrammide kategooriad** | Jah | Jah | Jah | Jah |
| **Saab lisada ühele hooldajale mitu klienti** | Ei | Ei | Ei | Ei |
| **Saab näha statistikat enim** | Ei | Ei | Ei | Ei |

| | | | | |
|---|---|---|---|---|
| **kasutatud piktogrammidest** | | | | |
| **Piltide vaade konfigureeritav vastavalt vajadustele** | Jah | Ei | Jah | Ei |

Alternatiivkommunikatsiooni rakendused, mis on eelnevalt välja toodud, kasutavad kõik andmete ja piltide salvestamiseks seadme lokaalselt andmebaasi. Lokaalse andmebaasi kasutamisel on palju eeliseid. Peamisteks eelisteks võib välja tuua järgnevad punktid [9].

- Puudub internetiühenduse vajadus [9].
- Andmete täiuslikkus on kasutaja vastutusel [9].
- Lokaalne andmete käsitlemine on märgatavalt kiirem [9].

Lokaalse seadme andmebaasi kasutamisel tekib ka miinuseid [9].

- Välisel hooldajal või tegevusjuhendajal puudub võimalus andmeid hallata ja omada ülevaadet [9].
- Andmetest võib kiiresti ilma jääda, kui kasutaja otsustab oma lokaalset andmeruumi tühjendada [9].

Välja toodud rakendused on omavahel väga sarnased ja peamised funktsionaalsused on identsed.

Esimese rakenduse Talk Up! [5] puhul saab välja tuua selge eelise, vaate konfigureeritavuse. Piltide suuruse seadistamine lubab hooldajal piltide suurust vastavalt kliendi vajadustele muuta. Samas jääb Talk Up! [5] rakenduse puhul suureks miinuseks piltide lisamise võimaluse puudumine.

Teise rakenduse JabTalk [6] juures on oluline see, et kasutajal on võimalik oma piktogrammide laud seadistada täpselt nende piktogrammidega, mis talle sobivad. JabTalk-i suur miinus on piltide suuruse seadistamisvõimaluse puudumine.

Kolmas, SymboTalk [7], sarnaneb funktsionaalsuste poolest esimesele rakendusele. SymboTalk [7] rakenduses on olemas nii piltide suuruse konfigureerimise võimalus kui ka võimalus lisada oma piktogramme.

Viimane rakendus PicCom [8] sarnaneb funktsionaalsuste poolest teisele rakendusele, kus mõlema puhul puudub võimalus piltide suurusi konfigureerida. Sellegipoolest on PicComil [8] võimalus lisada oma piktogramme vastavalt vajadusele.

Kõikide rakenduste puhul jääb suureks murekohaks piktogrammidega koostatud lausete edastamise võimaluse puudumine. Puuduse tõttu peab tegevusjuhendaja lause kuulmiseks asuma kliendi kõrval. Kui tegevusjuhendajale on määratud mitmeid kliente, siis on puudus märgatav, kuna tegevusjuhendajal ei ole võimalik mitme kliendiga korraga suhelda.

Tabelis 1 välja toodud funktsionaalsuste hulgast võib veel oluliseks puuduseks pidada piktogrammide kasutamise statistika puudumist. Hooldajana on oluline näha uute piktogrammide või piktogrammide suuruste muutumise mõju. Statistikata võivad piktogrammid, mida ei kasutata jääda märkamata ja sellega kitsendada kliendi eneseväljenduse skoopi.

Eelnevalt mainitud rakenduste tugev külg on funktsionaalsus, mille abil on võimalik luua sisukaid lauseid kasutades vaid piktogramme.

## 2.3 Vajadus uue rakenduse järgi

Eelnevas alampeatükis tehtud analüüsist selgub, et uurimistöös püstitatud probleemi, mis seisneb puudega inimese ja hooldaja vahelise distantsilt piktogrammidega suhtlemise funktsionaalsusega, turul hetkel ei pakuta.

Rakenduse nõuete kaardistamiseks viidi läbi intervjuu kolme puuetega inimeste hoolekande kodus töötajaga, kes tegelevad igapäevaselt puuetega inimeste tegevuste juhendamisega ja tegevusjuhendajate tegevuste juhendamisega.

Intervjuu läbiviimisel selgus, et rakenduses on vaja eristada kasutajaid erinevate rollide järgi. Nendeks rollideks on: klienditööjuht, tiimijuht, tegevusjuhendaja ja puudega inimene. Klienditööjuhi ja tiimijuhi vastutusaladeks on tegevusjuhendajate töö planeerimine ja juhendamine, ning uute puuetega inimeste majadesse määramine.

Tegevusjuhendaja vastutusala on puuetega inimeste igapäevaste tegevuste juhendamine ja nende abistamine.

Kuna klienditööjuhi ja tiimijuhi vastutusalad on mõningal määral kattuvad, otsustas autor ametikohad grupeerida süsteemi tasemel üheks, selleks rolliks on oragnisatsiooni juht. Rakenduses ei ole vajadust ka eristada tegevusjuhendajat, kes hetkel majas töötab, vaid tahvelarvutit, mis on klientide tegevuste juhendamiseks kasutusel, seega loodi roll maja. Hoolekande organisatsiooni skoobis kasutatakse puudega inimese asemel sõna klient, nii loodi ka roll klient.

Autor jagas tellija poolsed nõuded vastavalt ametikoha spetsiifilistele rollidele kolmeks: organisatsiooni juhi, maja ja kliendi nõuded. Nõuded kaardistatakse funktsionaalseteks ja mittefunktsionaalseteks nõueteks. Mõlemaid gruppe vaadatakse nii tellija kui ka arendaja vaatepunktist. Organisatsiooni juhi poolsed funktsionaalsed nõuded on grupeeritud nõude vastutusala järgi. Maja funktsionaalsed nõuded on seotud klientide tegevustega.

Organisatsiooni juhi nõuded.

- Organisatsiooni juhi vaates majaga seotud funktsionaalne nõue.
    - Maja lisamise ja redigeerimise funktsionaalsus.
- Organisatsiooni juhi vaates klienditööga seotud funktsionaalsused.
    - Kliendi lisamise ja redigeerimise funktsionaalsus.
    - Kliendile piktogrammide määramise funktsionaalsus.
    - Kliendile piktogrammi suuruse määramise funktsionaalsus.
    - Piktogrammide rakendusse üleslaadimise funktsionaalsus.
    - Piktogrammidele kategooria lisamise funktsionaalsus.
    - Kliendi piktogrammide kasutamise statistika vaatamise funktsionaalsus.

Rakenduses oleva rolli maja funktsionaalsed nõuded.

- Kliendi saadetud piktogrammide vaatamise võimalus.
- Kliendi sõnumi saamisel teate kuvamine.
- Kliendi teavitamine sõnumi kättesaamisest.
- Kõikide majale määratud klientide loendi vaatamine.

Rakenduses oleva rolli kliendi nõuded.

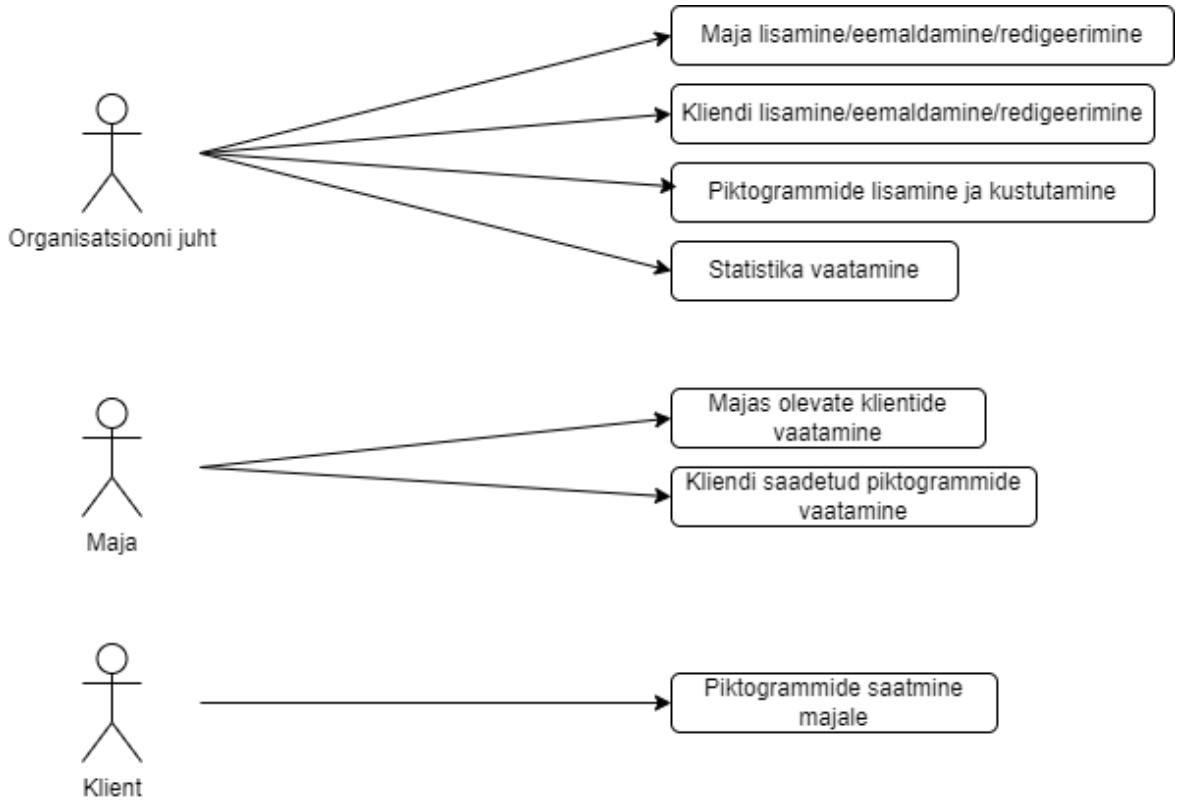
- Funktsionaalsed nõuded.
    - Piktogrammi saatmise funktsioon.
- Piktogrammide nägemise funktsionaalsus.
    - Kohale toimetatud sõnumi teavituse funktsioon.
    - Piktogrammile vajutades piktogrammi kohta helisignaali esitamine.
    - Mittefunktsionaalsed nõue.
    - Vaadetes ei tohi olla müra (võimalikult vähe funktsionaalsusi).

Rakenduse üldise kasutaja vajadustega seotud nõuded:

- Funktsionaalne nõue.
    - Sisse ja välja logimise funktsionaalsus.
- Mittefunktsionaalsed nõuded.
    - Kliendi-maja vahelise vestluse krüpteerimine.
    - Vaated peavad olema eestikeelsed.
    - Rakendus peab olema kasutatav erinevatel operatsioonisüsteemidel, näiteks Android, Windows, IOS.

Mittefunktsionaalsed nõuded rakendusele arendajale.

- Kood peab olema kirjutatud inglise keeles.
- Kood peab olema kirjutad puhta koodi reeglite järgi.
- Kood peab olema kooskõlas ärilise keelega.
- Kood peab olema välises repositooriumis.
- Ärilised funktsionaalsused peavad olema põhjalikult testitud.
- Kood peab olema kirjutatud piisavalt sõltumatult, selleks et tulevikus oleks võimalik lihtsasti kasutajaliidest või andmebaase välja vahetada.




Joonis 1 Rollide kasutusloo diagramm

Joonisel 1 on toodud välja võimalikud funktsionaalsused, mida selles rollis kasutaja saab teha. Diagrammil on kolm rolli organisatsiooni juht, maja ja klient. Nende funktsionaalsused on kirja pandud eelnevalt määratletud nõuete alusel.

## 2.4 Arenduskeskkonnad

Kasutajaliidese arendamiseks kasutatakse keskkonda *Visual Studio Code* [10]. Keskkonna valimisel lähtus autor varasemast kogemusest ja vajalike funktsionaalsuste kooslusest. Kuna autor oli varasemalt *Visual Studio Code*i kasutanud, oli valik lihtne. Platvorm on avatud lähtekoodiga ja koodi kirjutamise lihtsustamiseks on teiste kasutajate poolt loodud palju erinevaid pistikprogramme.

*Back-end* koodi integreeritud arendamise keskkonna (inglise keeles *integrated development environment*, edaspidi ka IDE) valikul oli autoril kaks valikut, *Visual Studio* ja *Jetbrains Rider*. Autor oli varem mõlemat IDEd kasutanud. *Rider*il on kõvasti mahukam võimekus koodi analüüsimisel ja vigade tuvastamisel [11]. Selle tõttu valis autor IDEks *Jetbrains Rider*i.

Väliste teenuste, pildibaasi ja rakenduse andmebaasi, lisamiseks kasutati Dockerit. Dockeriga saab luua mugavalt eraldiseisvaid teenuseid Docker-i konteinerites [12]. Dockeri konteinerites on andmebaasina kasutatud PostgreSQLi ja pildibaasina MinIO teenuseid.

# 3 Rakenduse arhitektuuri loomine

Peatükis antakse ülevaade uue süsteemi arhitektuurist. Nõuete alusel luuakse rakenduse arhitektuurile ülevaatlikud diagrammid ja valitakse vastavalt arhitektuurile ka sobilikud programmeerimiskeeled ning tarkvaraarenduse raamistikud. Rakenduse arhitektuuri loomisel uuriti hoolekandekodu vajadusi ja nende kasutusel olevaid seadmeid.

Bakalaureusetöö rakenduse struktuuri loomiseks valis autor neljakihilise arhitektuuri. Valiku tegemisel lähtus autor nõudest, et iga kiht peab olema lihtsasti vahetatav. Seda ei saa tagada kolmekihilise arhitektuuri puhul, kus kasutajaliides ja API on omavahel ühendatud [13].

Neli kihti on järgnevad:

- kasutajaliides,
- API,
- domeen ja
- infrastruktuur.

Kasutajaliidese kiht vastutab otseselt kasutajale informatsiooni näitamise ja käskude süsteemi saatmise eest. API vastutab kasutaja päringute ja käskude delegeerimise eest vastavatesse rakenduse loogika kihi osadesse. Domeen vastutab ärilise mudeli kontseptsioonide, selle kohta käiva informatsiooni ja kõikide äriliste reeglite hoidmise eest. Infrastruktuuri kiht vastutab kogu rakenduse omavahelise suhtluse eest, milles on nii andmebaasiga suhtlus kui ka piktogrammide hoidmise ja töötlemise teenusega. [14]

Diagrammide loomisel on kasutatud internetipõhist ühendatud modelleerimise keeles (inglise keeles *Unified Modeling Language*, edaspidi ka UML) diagrammide loomise rakendust *Draw.Io* [15].

UML on kogum diagrammidest, mille kasutamisel on lihtne anda edasi rakenduse spetsiifikaid ja üleüldist ülesehitust. [16]

## 3.1 Kasutajaliides

Läbiviidud intervjuust selgus, et kasutajatel on peamiselt kasutusel Androidi operatsioonisüsteemil toimivad tahvelarvutid. Organisatsiooni juhid kasutavad aga igapäevatöös arvuteid. Seega pidi kasutajaliidese raamistiku valikul leidma lahenduse, et see töötaks nii veebis kui ka nutiseadmes. Parim lahendus, et tagada veebirakenduse ja nutiseadme rakendusega, on luua kasutajaliides kasutades progressiivse veebirakenduse (edaspidi ka PWA ehk inglise keeles *Progressive Web Application*) lahendust. [17]

PWA on veebirakendus, mille peamine eesmärk on tagada väga sarnast kogemust seadmespetsiifilise operatsioonisüsteemi peale kirjutatud rakendustega. Progressiivse veebirakenduse eelised on järgmised [17].

- Progressiivne - töötab igas brauseris [17].
- Kohanduv - sobib igale seadmele [17].
- Sõltumatus veebiühendusest - *cache*-imise abil saab rakenduse funktsionaalsusi kasutada ka halva internetiühenduse korral [17].
- Installeerimine - kasutajatel on lihtne veebirakendust oma seadmesse installeerida rakenduse poode kasutamata [17].
- Lihtsasti leitavad - tänu W3C manifestidele on rakendus optimeeritud ka otsingumootoritele [17].
- Kaasahaaravad - kuna veebirakendus töötab sarnaselt seadme spetsiifilisele rakendusele, siis on lihtne saata teavitusi, kasutades mõnda sellist teenust [17].
- Turvaline - PWA-sid serveeritakse HTTPS protokolli kasutades [17].

Autor otsustas raamistike valikul lähtuda oma eelnevatest kogemustest. Seega valis autor sobivaks raamistikuks Vue.Js, sest seda on autor kõige rohkem kasutanud.

Vue.Js on JavaScripti raamistik, mida kasutatakse kasutajaliidese arendamiseks. Raamistik annab võimaluse luua kasutajaliidest kasutades komponendipõhist lähenemist [18]. Vue.Js raamistikuga on võimalik kiiresti kasutajaliidese kihti luua kasutades käsurea liidest (edaspidi ka CLI). Kasutades CLI-d saab arendaja konfigureerida rakendust täpselt oma vajaduste järgi [19]. Kasutajaliidese projekti arendust alustas

autor kasutades Vue.Js PWA *template*i, mida saab valida CLI projekti konfigureerimise vaatest.

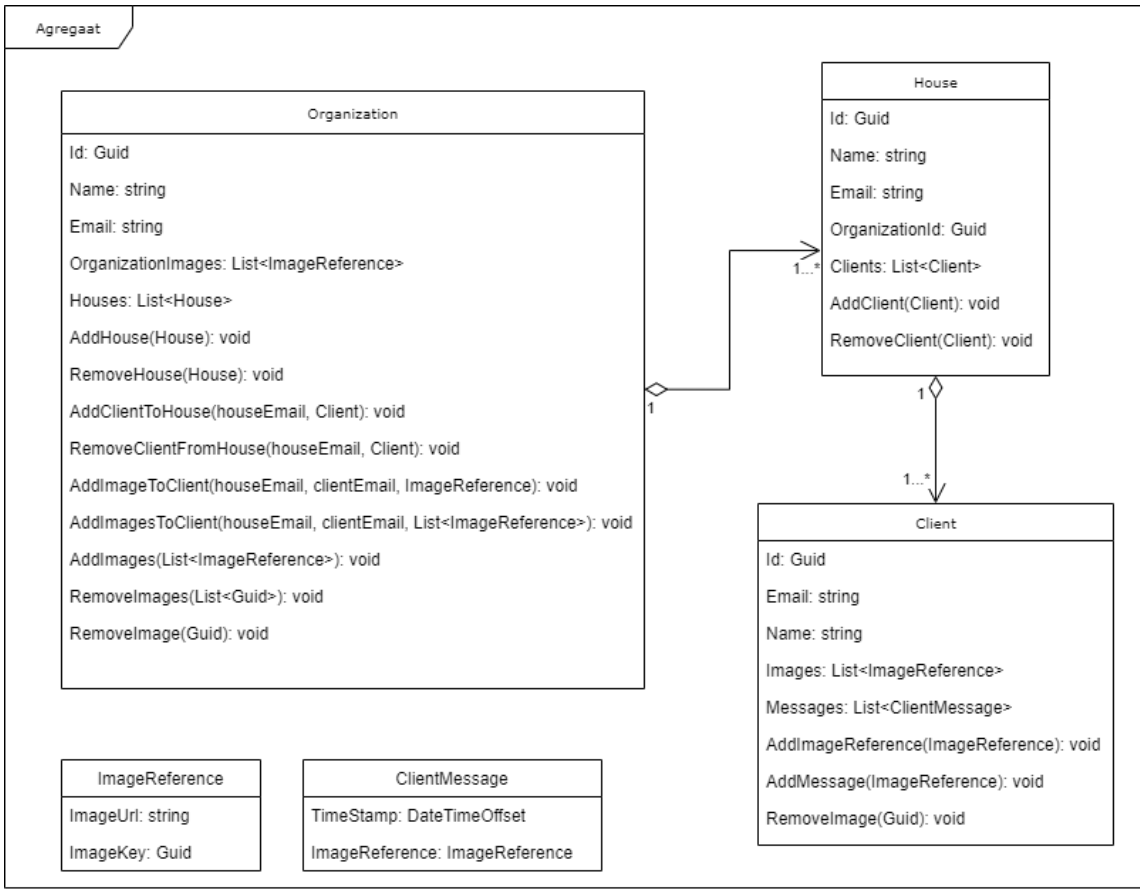
## 3.2 Äriloogika kiht

Antud peatükis keskendutakse äriloogika kihile, seal olevatele mudelitele ja mudelitele seatud piirangutele. Domeeni mudeli loomisel kaasati protsessi inimesed, kes tegelevad igapäevased puuetega inimeste tegevuste juhendamisega ja tegevusjuhendajate tegevuste juhendamisega.

Intervjuudest selgus, et üldine organisatsiooni ülesehitus koosneb organisatsiooni juhtidest, tegevusjuhendajatest ja puuetega inimestest. Reaalses elus on hoolekande teenuse pakkumise protsess jagunenud kolmeks. Klienditöö juht ja tiimitöö juht, tegevusjuhendaja ja klient. Rakenduses on nimetatud tegevusjuhendaja järgi olem *House*. Kus üks maja representeerib kogumit klientidest. Majas olevad puudega inimesed on rakenduses näidatud olemina *Client*. Nimetuste loomisel lähtuti domeeni juhitud programmeerimise reeglitele, kus on öeldud, et äriloogika koodibaas peab rääkima ärikeelt [14].

Domeen on osa tarkvarast, mis tegeleb täpsete äriliste probleemide lahendamisega. Parima äriloogika kihi loomiseks on oluline, et see oleks teistest rakenduse sõltuvustest ja tehnilisest loogikast eraldatud. Eraldatavuse tagamiseks on lihtsaim viis viia kogu äriloogika rakenduses eraldi kihti. [14]

Äriloogika kihis eraldatavuse tagamiseks kasutatakse arendamisel kihilist arhitektuuri. Kihilise arhitektuuri peamine väärtus tuleneb sellest, et iga spetsiifilise osa eest programmis vastutab spetsiifiline kiht [14].




Joonis 2 Domeeni mudel

Domeeni mudeli loomisel on kasutatud mitmeid mudeli juhitud arenduse mustreid. Peamised mustrid, mis jooniselt 2 välja paistavad, on agregaadi muster, olemite muster ja väärtusobjektide muster.

Agregaat on abstraktsioon, millega kapsuleeritakse mudelis olevaid sõltuvusi. Agregaadis on mitmeid olemeid, mis on ühendatud üheks. Agregaadi juur on üks olem, mis on ainsana agregaadi piiridest väljas asuvatele objektidele kättesaadav ja mida võivad välised objektid hoida. Kuna agregaadi puhul toimub muteerimine vaid juure kaudu, siis on lihtne tagada, et piiritletud invariandid on alati tagatud, ning kõik agregaadi ühendis olevad olemid on valiidsed. [14]

Olem on objekt, millel on kindel identiteet, mille muud väärtused võivad aja jooksul muutuda, aga identiteet jääb alati samaks. Olemeid kutsutakse ka viite objektideks. Agregaadi juurel peab olema globaalne identiteet, kuna seda on vaja eristada üle rakenduse. Piirides olevatel olemite vahel ei ole aga vaja eristada olemeid väljaspool agregaadi piire, selle tõttu on nendel olemitel lokaalne identiteet. [20]

Antud rakenduses on määratud olemiteks *Organization*, *House* ja *Client* (vt. ka joonis 2). *Organization*il, on globaalne identiteet. See tähendab, et meil on vaja organisatsioone üle terve süsteemi üksteisest eristada [14]. Olemitel *House* ja *Client* on aga lokaalne identiteet, mis tähendab, et olemeid on vaja eristada üksteisest vaid agregaadi piirides, üle terve süsteemi eristamine ei ole antud kontekstis nõutud [14].

Väärtusobjekt on muteerimatu atribuutide kogum, milles hoitakse mingi olemi atribuute ja selle piire [21]. Antud rakenduses on loodud kaks väärtus objekti *ImageReference* ja *ClientMessage*. Väärtus objektis *ImageReference* on atribuudid, mis on seotud pildibaasis hoiustatud andmetega. Väärtus objektis *ClientMessage* hoiustatakse andmeid kliendi poolt saadetud sõnumi kohta, aeg millal saadeti ja *ImageReference* piktogrammist, mis saadeti.

Joonisel 2 on näha, et agregaadi juureks on valitud olem *Organization*. *Organization*i olemis on omakorda määratud agregaadi piirid. Agregaadi piirides on olemid *House* ja *Client*.

Agregaadi juur *Organization* hõlmab endas ühe terviku asutuse andmeid. Asutuses võib olla mitmeid maju, mitmetes majades võib olla mitmeid kliente. Oluline on, et kõik muudatused käiksid kasutades agregaadi juurt *Organization*. Asutuse olemit ja selle agregaadi piirides olevaid maju, kliente ning pilte saab muteerida kasutades järgnevaid meetodeid:

- *AddHouse* - tegutseb majade lisamisega;
- *RemoveHouse* - vastutab majade eemaldamise eest;
- *AddClient* - vastutab majja kliendi lisamise eest;
- *RemoveClient* - vastutab majast kliendi eemaldamise eest;
- *AddImageToClient* - vastutab majas oleva kliendi piktogrammide lisamise eest;
- *RemoveImageFromClient* - vastutab majas oleva kliendi piktogrammide eemaldamise eest;
- *ChangeClientViewDimensions* - vastutab majas oleva kliendi piltide arvu ekraanil muutmise eest;
- *AddImageReference* - vastutab organisatsiooni pildibaasi muteerimise eest;
- *AddImageReferences* - vastutab organisatsiooni pildibaasi muteerimise eest;
- *RemoveImageReference* - vastutab organisatsiooni pildibaasi muteerimise eest.

### 3.2.1 Invariandid

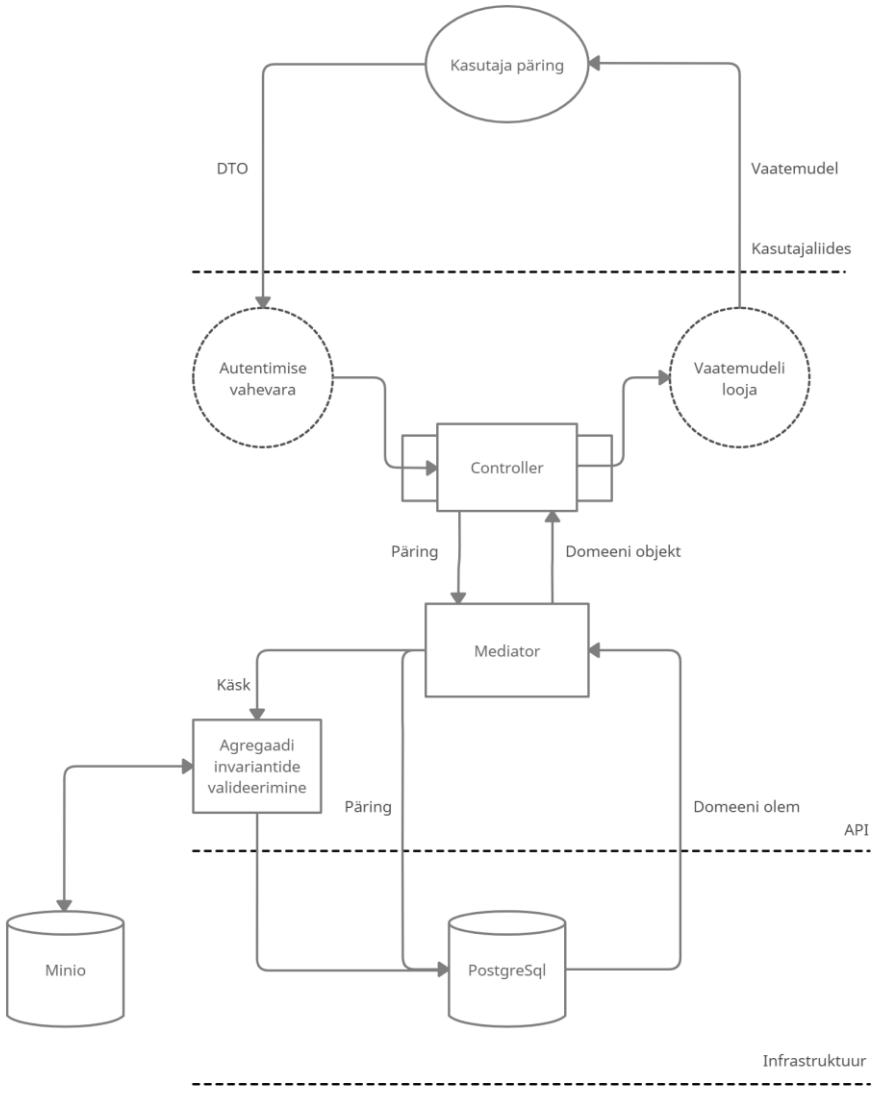
Agregaadi invariandid on kogum ärireeglitest, mille korrektsel kehtestamisel on agregaadi olek alati korrektne. Invariante kehtestatakse agregaadi juures enne igat muteerimist ja iga muteerimise ajal. [14]

Agregaadis *Organization* on määratletud järgnevad invariandid:

- organisatsiooni nimi ja email ei tohi olla tühjad;
- maja email, nimi ja organisatsiooni identifikaator ei tohi olla tühjad;
- kliendi nimi ja email ei tohi olla tühjad;
- kliendile pildi lisamisel peab sama pilt olema olemas ka organisatsiooni pildibaasis;
- organisatsiooni pildibaasist pildi eemaldamisel peab sama pildi eemaldama ka kõikidest klientidele määratud piltidest.

## 3.3 API ja infrastruktuur

Alampeatükis kirjeldatakse rakenduse API ja infrastruktuuri kihtide koostööd ja rakenduse suhtlust. API Kihis suunatakse kõik päringud ja käsud õigesse vastutusalasse. Infrastruktuuri kihis määratakse ära tehnilised spetsiifikad [14]. Kihis on loodud funktsionaalsused, millega saab andmebaasis olevaid olemeid muteerida ja pärida. Samuti suhtleb kiht väliste teenustega. [14]




Joonis 3 Süsteemi kommunikeerimise diagramm

### 3.3.1 API

API kihi loomisel kasutas autor mudel-vaade-kontroller (edaspidi ka MVC) mustrit. MVC mustris kirjeldatakse rakenduse API kihi arhitektuuri kolmes osas: mudel, vaade ja kontroller. Mudel on rakenduses domeenispetsiifiline simulatsioon või implementatsioon äriloogika mudeli struktuurist. Vaade on kasutajale kuvatav graafiline visuaal mudelist. Kontrollerid on liideseks kasutaja ja rakenduse vahel. Kontrollerid delegeerivad kasutajate käske erinevatesse vastutusaladesse. Kontrolleritesse saabuvad vaatest kasutaja käsud ja päringud ning kontroller tagastab kasutajale vaateid. [22]

Läbiviidud intervjuudest tuli välja kolm kasutajate gruppi. Kasutajad, kes on organisatsiooni juhatajad, tegevusjuhendajad ja kliendid. Sellele vastavalt on määratud rakenduses kolm erinevat kontrollerit. Kontrolleriga määratakse ära kasutaja

vastutusalad ja igal kontrolleril määratud nõudmine, et seda saab kasutada vaid õiges rollis olev isik.


Joonis 4 Kontrollerid ja DTO-d

Organisatsiooni kasutajate rollid on määratud kooskõlas intervjueeritavate informatsiooniga. Süsteemis on olemas kolm rolli:

- organisatsiooni juht;
- maja;
- klient.

Igale rollile on omakorda jaotatud kasutusõigused, organisatsiooni juhi õigused jagunevad järgnevalt:

- saab vaadata kliente;
- saab vaadata maju;

- saab vaadata klientide sõnumeid;
- saab vaadata organisatsiooni pildibaasi;
- saab laadida üles uusi pilte;
- saab lisada uue maja;
- saab lisada uue kliendi.

Maja õigused on määratud järgnevalt:

- saab vaadata kliente;
- saab vaadata klientide sõnumeid.

Kliendi õigused on määratud järgnevalt:

- saab saata sõnumeid.

Joonisel 4 on näidatud kõik rakenduses olevad kontrollerid. *OrganizationsController*-i otspunkte on lubatud kasutada vaid organisatsiooni juhi rollis olevatel kasutajatel. *HouseControlle*ri otspunkte on õigus kasutada vaid tegevusjuhendaja rollis oleval kasutajal ja *ClientsControlle*ri otspunkte on õigus kasutada vaid kliendi rollis oleval kasutajal.

Joonisel 4 on märgitud ära ka iga kontrolleri kohta selle otspunktides kasutatavad andmete liigutamise objektid (inglise keeles *Data Transfer Object*, ehk DTO). DTO hoiab endas andmeid, mis tulevad kas äriloogika kihist või kliendi päringute näol. DTOd aitavad vähendada üleliigset infot, mis võib ilma DTOde kasutamiseta kasutajale lekkida. Samuti pannakse DTOde kasutamisega paika kindel skeem, mille alusel on võimalik kontrolleri otspunkti päringuid ja käske saata. [23]

Organisatsioonide kontrolleris on kasutsel viis DTOd. Nendeks on *OrganizationViewModel*, *HouseViewModel*, *ClientViewModel*, *HousePostModel* ja *ClientPostModel*. Majade kontrolleris on kasutusel *HouseViewModel* ja *ClientViewModel*. Klientide kontrolleris on kasutusel *ClientViewModel*.

### 3.3.2 Infrastruktuur

Infrastruktuuri kihis on rakendusevälise ja -sisemise ühendamisega seotud tehnilised meetodid ja raamistikud. Kihi peamine eesmärk on hoida suhtlust erinevate kihtide vahel efektiivse ja lihtsana. [14]

Antud kihis on loodud andmebaasiga suhtlemise funktsionaalsus kasutades päringute ja käskude vastutusalade segregeerimise printsiipi. Päringute ja käskude vastutusalade segregeerimise (inglise keeles *Command Query Responsibility Segregation*, edaspidi ka CQRS) printsiip jagab andmebaasipäringud kahte gruppi. Päringud, mis vastutavad andmete lugemise eest ja käsud, mis vastutavad andmebaasi objektide salvestamise ja muteerimise eest. CQRSi peamine eelis on see, et iga päring toimub erineva mudeliga. [24]

Käskude ja päringute saatmiseks on kasutatud MediatR [25] raamistikku, mille abil on lihtne luua iga käsu ja päringu jaoks oma mudel.

Rakenduse andmebaas on üles seatud kasutades PostgreSQLi. Andmebaasiskeemide loomiseks on kasutatud MartenDBd millega luuakse dokumentidena struktureeritud andmebaasi objektid [26].

Dokumentidena salvestatud andmebaasi objektid sobivad kokku eelnevalt kirjeldatud agregaadi mustriga. Hea sobivus saavutatakse, sest andmebaasist pärimine toimub dokumentide kaupa. Sarnaselt agregaatidele, hoitakse ühes dokumendis kogu informatsiooni selle dokumendi kohta. [27]

Infrastruktuuri kihis on ka olemas võimekus suhelda pildibaasi teenusega. Piltide salvestamiseks ja pärimiseks on kasutatud MinIO teenust. MinIO on objektide hoiustamise süsteem [28].

# 4 Arendusmetoodika ja valminud kasutajaliides

## 4.1 Arendusmetoodika

Kogu koodi kirjutamise vältel lähtus autor puhta koodi printsiipidest. Kood on puhas, kui selle eest on kantud hoolt, selle lugemine on sarnane raamatu lugemisele. Lugemisel on lihtne visualiseerida kogu süsteemi tööahelat. [29]

Tagarakenduse (inglise keeles ka *backend*) koodi kirjutamisel lähtus autor testide-juhitud arenduse (inglise keeles *Test-Driven Development*) metoodikast. Metoodika on tarkvara arendamise teguviis, kus esmalt kirjutatakse testid, seejärel minimaalne funktsionaalne kood testi läbimiseks, seejärel refaktoreeritakse kirjutatud koodi. Protsessi järgimisel kirjutatakse esmalt liides ja meetodite signatuurid, sedasi programmeerides väheneb üleliigse koodi hulk. [30]

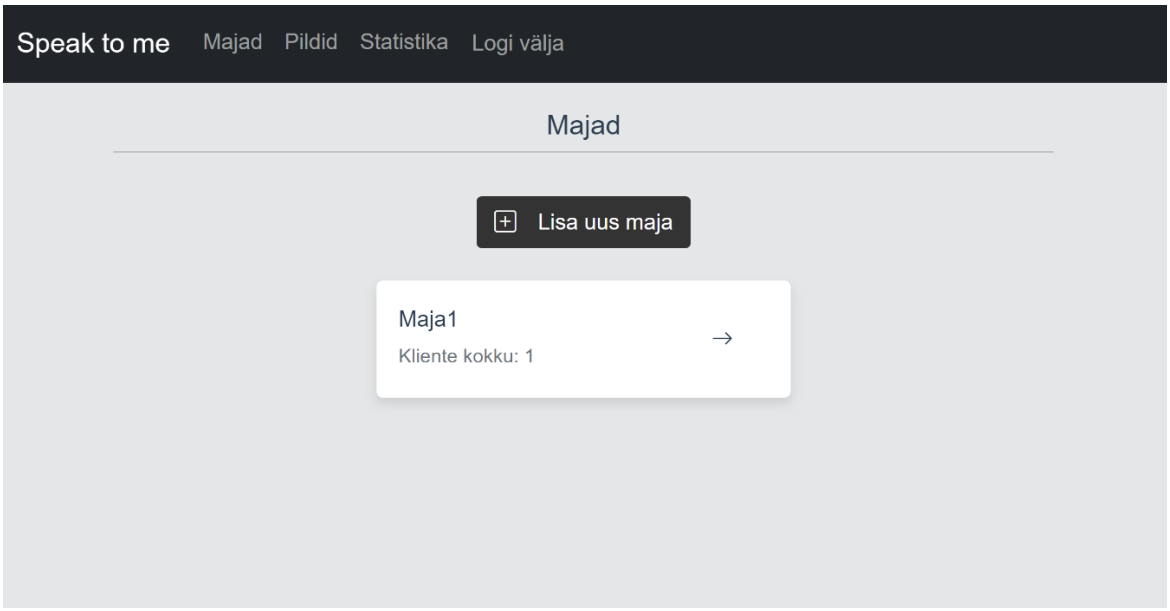
Testid loodi tagarakenduse koodibaasi iga funktsionaalsuse jaoks. Loogika testimisel kasutati integratsiooniteste ja ühikteste. Integratsioonitestideks loodi uus andmebaas, mille vastu tehti päringuid ja käske. Ühiktestide loomiseks kasutati NUnit raamistikku ja Moq raamistikku.

## 4.2 Valminud rakenduse kasutajaliidese näited

Rakenduse arendamisel lähtus autor eelnevalt loodud arhitektuurist. Sellest selgus, et on kolm vaadet, mida kasutajad vajavad - organisatsiooni juhi, maja ja kliendi vaated. Bakalaureusetöö mahtu arvestades ei ole võimalik kõiki funktsionaalsuseid eraldi lahti kirjutada, seega on peatükis pildid ja kirjeldused vaid kasutajaliidese kohta. Täpsema koodi vaatamiseks on võimalik vaadata seda avalikust *Github*i repositooriumist.

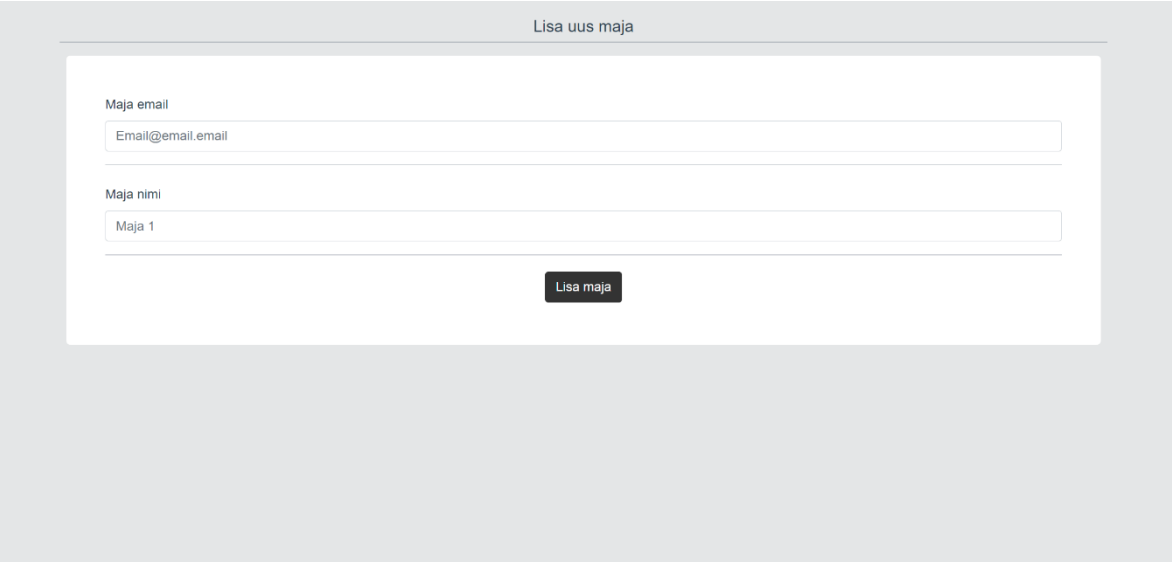
### 4.2.1 Organisatsiooni juhi vaated

Alampeatükis on näidatud pilte organisatsiooni juhi vaadetest. Kogu tagarakenduse loogikaga suhtlus toimub läbi *OrganizationControlleri*. Kontrolleris on kõik otspunktid, mida kasutatakse kasutajaliideses andmete kuvamiseks ja muteerimiseks.



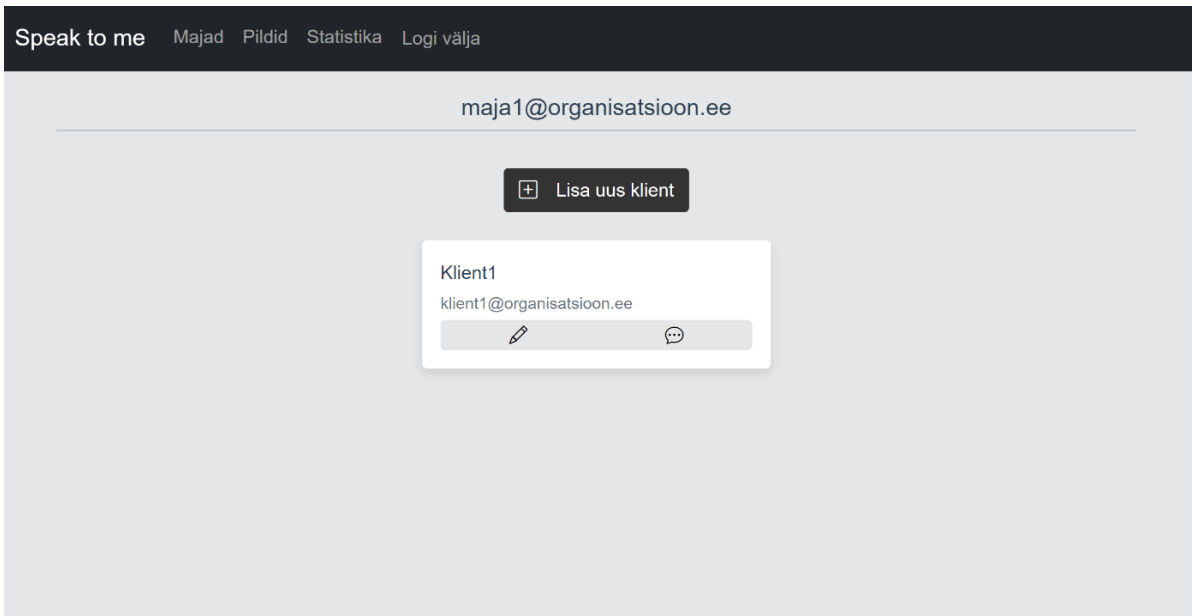

Joonis 5 Organisatsiooni majade loendivaade

Joonisel 5 on kujutatud organisatsiooni juhi kasutaja majade loendivaadet. Vaates on kasutajal võimalik vaadata kõiki tema organisatsiooni kuuluvaid maju ning lisada uusi maju. Vajutades nupule „Lisa uus maja“ suunatakse kasutaja edasi majade lisamise vaatesse. Majale vajutades suunatakse kasutaja edasi klientide loendivaatesse. Ebaõnnestunud lisamise korral teavitatakse kasutajat vastavate veateadetega.



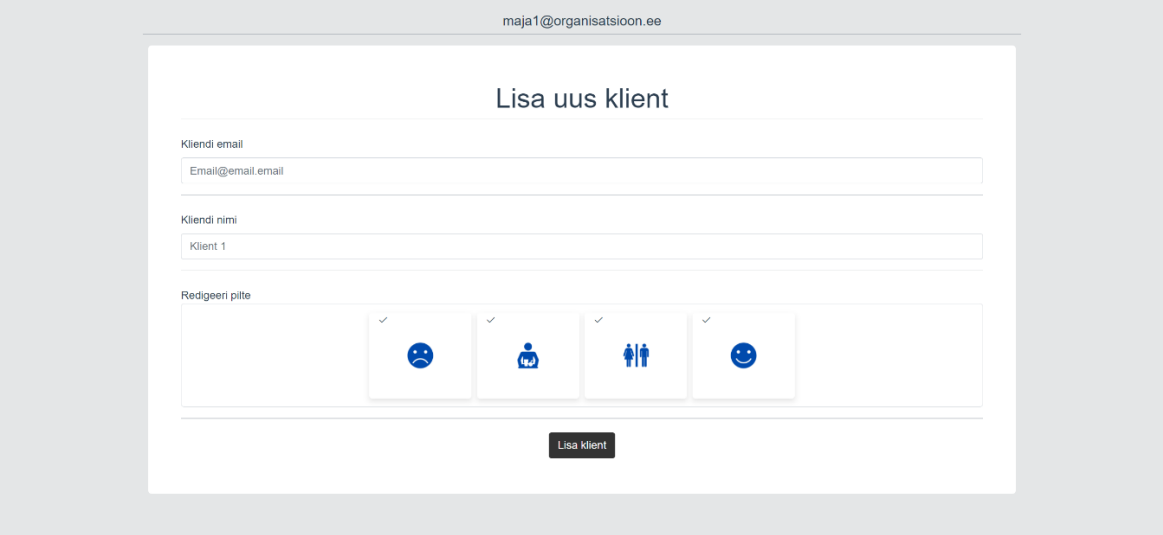

Joonis 6 Organisatsiooni maja lisamise vaade

Joonisel 6 on kujutaud maja lisamise vaade. Kasutajal võimalik sisestada email ja maja nimetus. Vajutades nupule Lisa maja saadetakse serverile päring uue maja lisamise kohta. Õnnestumisel suunatakse tagasi majade nimekirja.



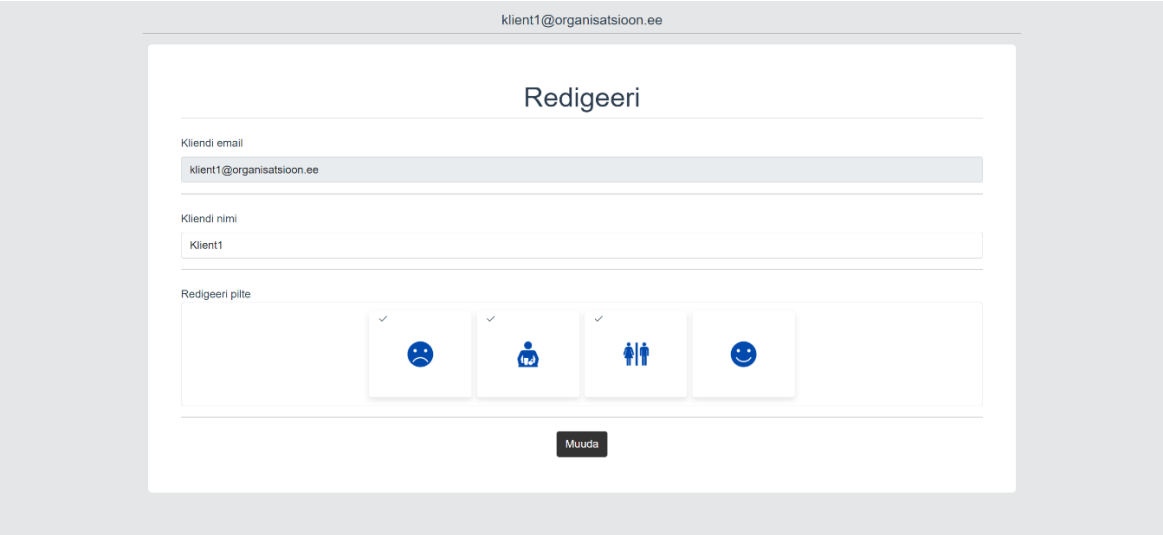

Joonis 7 Organisatsiooni klientide loend

Joonisel 7 on näha klientide loendivaadet. Klientide vaates kuvatakse majas olevaid kliente. Klientide nimekirja on võimalik lisada uusi kliente vajutades nupule „Lisa uus klient“. Nupule vajutades suunatakse kasutaja edasi kliendi lisamise. Igal kliendil on ka kaks nuppu, mis vasakult paremale järjestuses on redigeerimine ja vestlus. Vajutades nupule „Redigeeri“ suunatakse kasutaja edasi kliendi redigeerimise vaatesse. Vajutades nupule „Vestlus“ suunatakse kasutaja edasi kliendi vestluse vaatesse.



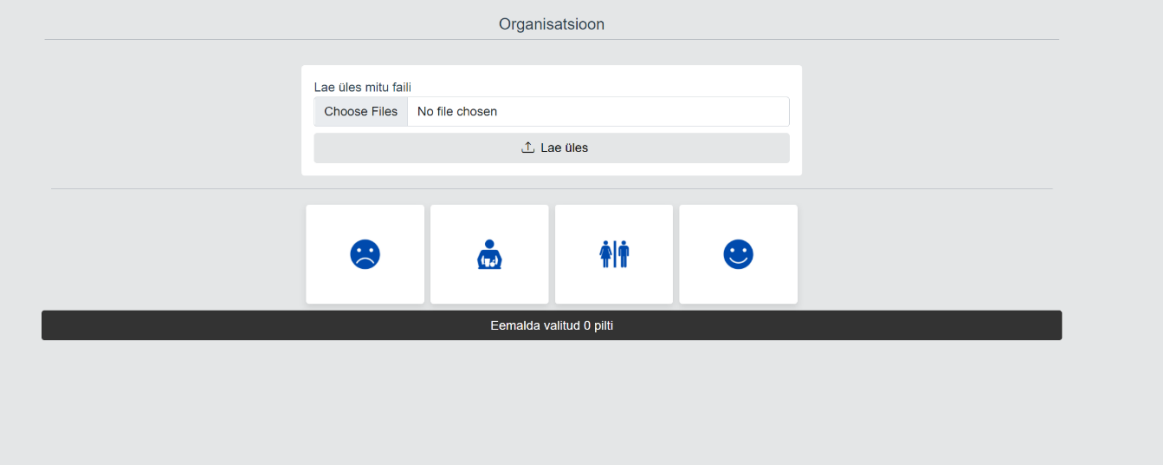

Joonis 8 Organisatsioon lisa uus klient vaade

Joonisel 8 on uue kliendi lisamise vaade. Uue kliendi lisamiseks on vaja täita kliendi emaili ja kliendi nime lünk. Seejärel on kuvatud kõik organisatsioonile kuuluvad pildid, millest peab kasutaja tegema valiku. Valitud pildid määratakse kliendile. Kliendi pilte kuvatakse kliendi piktogrammide vaates. Ebaõnnestumise korral kuvatakse kasutajale tagasiside, mis läks valesti.




Joonis 9 Organisatsioon redigeeri kliendi vaade

Joonisel 9 on kujutatud kliendi redigeerimise vaade. Kliendi redigeerimise vaates on võimalik muuta kliendi nime ja talle määratud pilte. Emaili muutmine ei ole võimaldatud.

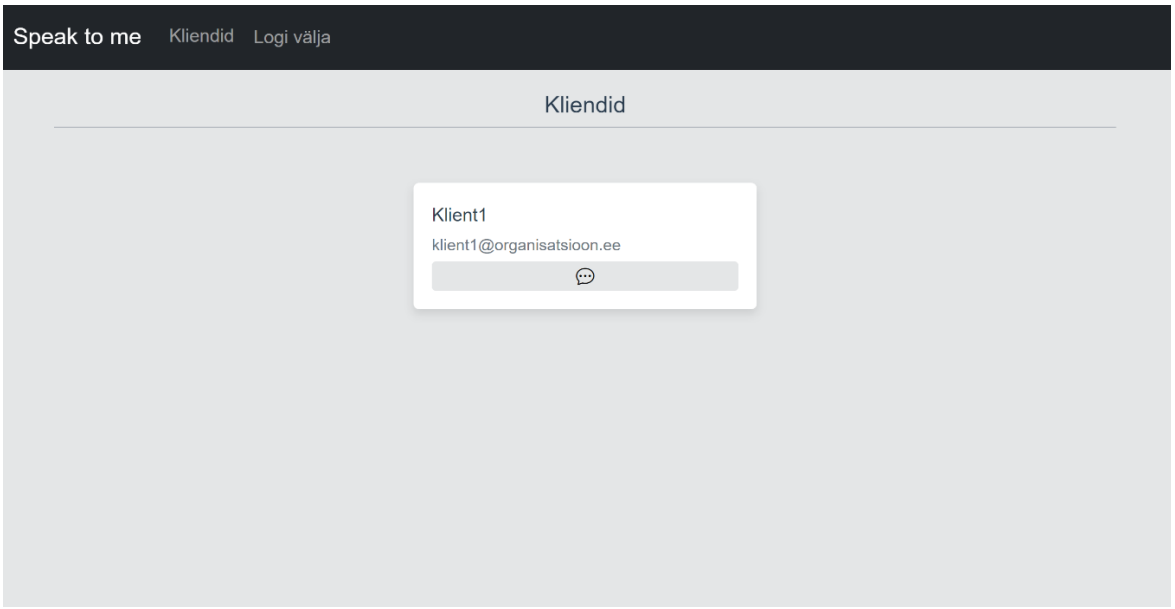



Joonis 10 Organisatsiooni piltide redigeerimise vaade

Joonisel 10 on kujutatud organisatsiooni piltide vaade. Pilte on võimalik rakendusse üles laadida vajutades nuppu „*Choose files*“ ja seejärel „Lae üles“. Õnnestunud üles laadimisel tekivad pildid alumisse piltide loendit. Ebaõnnestunud üles laadimise korral kuvatakse veateade.

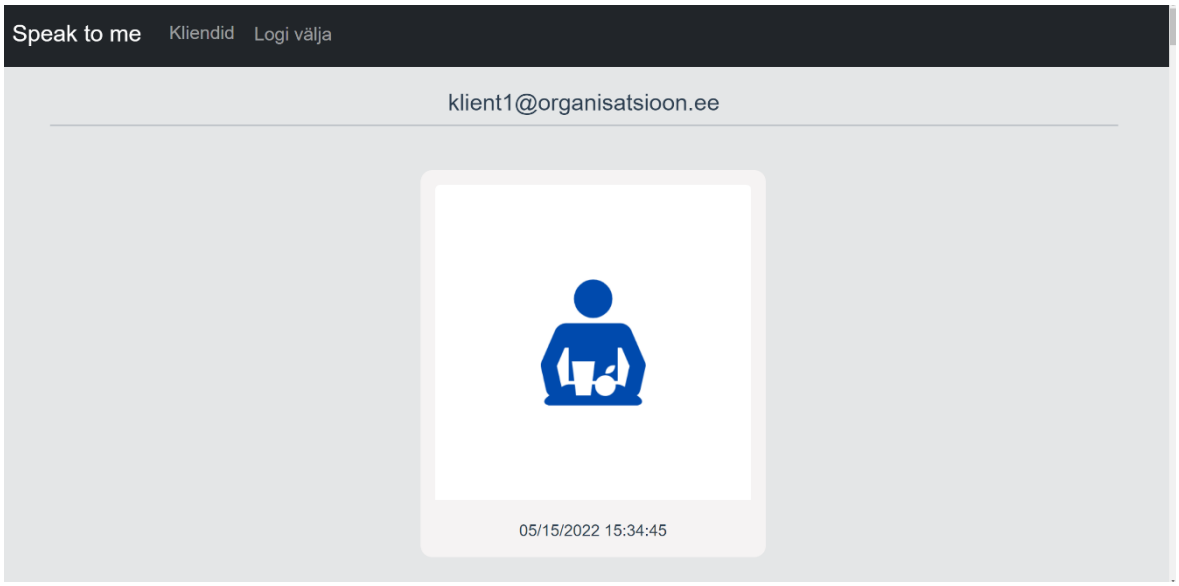
### 4.2.2 Maja vaade

Maja vaated on kuvatud maja rollis olevale kasutajale. Kõik maja vaatega seotud otspunktid asuvad failis *HouseController*.




Joonis 11 Maja klientide listivaade

Joonisel 11 on kuvatud maja kasutaja klientide listivaadet. Klientide vaates on võimalik kõiki majja määratud kliente sirvida ja nende vestlust avada. Kliendi listi elemendi peal on nupp „Vestlus“. Vajutades nupule „Vestlus“ avatakse kasutajale vaade kliendi saadetud sõnumitest.

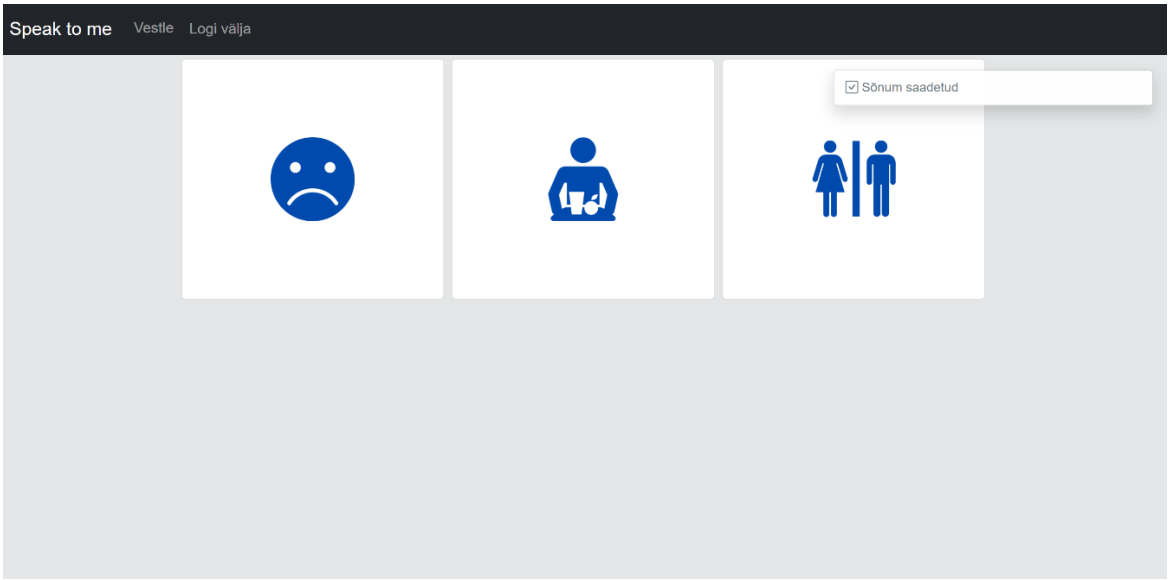



Joonis 12 Kliendi vestluse vaade

Joonisel 12 on näha maja kasutaja kliendiga vestluse vaadet. Vaates kuvatakse kõiki temale saabunud sõnumeid, mis on järjestatud nii, et uuemad sõnumid on esimesed.

### 4.2.3 Kliendi vaade

Kliendi vaade avatakse kasutajale, kui õnnestunud sisselogimise korral on talle määratud roll klient. Vaated kasutavad kontrollerit *ClientController*.




Joonis 13 Kliendi piktogrammide vaade

Joonisel 13 on näha kliendi piktogrammide vaadet. Piktogrammide vaates on kasutajale kuvatud kõik temale määratud piktogrammid. Piktogrammile vajutades saadetakse sõnum majale, õnnestunud saatmise korral tuleb üleval paremal teade „Sõnum saadetud“.

## 4.3 Järeldus

Lõputöö raames arendati välja minimaalne kasutatav rakendus, millega on lihtne luua vestluskanal kliendi ja tegevusjuhendaja vahel. Rakenduse arhitektuurile lähtudes, kus agregaadis on organisatsiooni andmed, majade andmed ja klientide andmed, on tarkvara on sobilik eelkõige hoolekandeteenustele, kus kliente ja tegevusjuhendajaid on palju.

Tänu PWA kasutamisele kasutajaliideses on rakendust võimalik kasutada sõltumata platvormist ja seadmest. Selle abil on organisatsiooni juhatusel ja tegevusjuhendajatel võimalik lihtsasti kliente sirvida ja redigeerida. Klientidel on võimalik mugavalt tahvelarvutist piktogramme saata.

Piiratud aega ja lõputöö mahtu arvestades suudeti täita 10 funktsionaalset nõuet 16st ja 9 mittefunktsionaalset nõuet 10st. Funktsionaalsetest nõuetest täideti arenduse jooksul ära järgnevad nõuded.

Organisatsiooni juhi nõuded, mis said realiseeritud.

- Majadega seotud funktsionaalsus.
    - Maja lisamise ja redigeerimise funktsionaalsus.
- Klienditööga seotud funktsionaalsused.
    - Kliendi lisamise ja redigeerimise funktsionaalsus.
    - Kliendile piktogrammide määramise funktsionaalsus.
    - Kliendile piktogrammi suuruse määramise funktsionaalsus..
- Piktogrammidega seotud funktsionaalsused.
    - Piktogrammide rakendusse üleslaadimise funktsionaalsus.

Maja nõuded, mis said realiseeritud.

- Kliendi saadetud piktogrammide vaatamise võimalus.
- Kõikide majale määratud klientide loendi vaatamine.

Kliendi nõuded, mis said realiseeritud.

- Piktogrammi saatmise funktsioon.
- Piktogrammide nägemise funktsionaalsus.
- Vaadetes ei tohi olla müra (võimalikult vähe funktsionaalsusi).

Üldise kasutaja nõuded, mis said realiseeritud.

- Sisse ja välja logimise funktsionaalsus.
- Vaated peavad olema eestikeelsed.
- Rakendus peab olema kasutatav erinevatel operatsioonisüsteemidel, näiteks Android, Windows, IOS.

Arendaja poolt esitatud mittefunktsionaalsed nõuded, mis said realiseeritud.

- Kood peab olema kirjutatud inglise keeles.
- Kood peab olema kirjutad puhta koodi reeglite järgi.
- Kood peab olema kooskõlas ärilise keelega.
- Kood peab olema välises repositooriumis.
- Ärilised funktsionaalsused peavad olema põhjalikult testitud.
- Kood peab olema kirjutatud piisavalt sõltumatult, selleks et tulevikus oleks võimalik lihtsasti kasutajaliidest või andmebaase välja vahetada.

Kuna arendamisel kasutati kaasaegseid tehnoloogiaid ja rafineerituid metoodikaid, siis tarkvara edasiarendamine on kiire ja ohutu. Kiire edasiarendamine on tagatud süsteemi infrastruktuuri ja äriloogika kihi heale ühesehitusele. Edasiarendamise ohutus tuleneb mahuka testituse abiga. Peamise loogika testituse abil on vana koodi refaktoreerimine lihtne, sest loogikat muutes annavad testid kohese tagasiside, kui mõni loogika peaks purunema.

## 4.4 Tagasiside hooldekodu töötajatelt

Lõputöö raames loodud tarkvara demonstreeriti ka hooldekodu tegevusjuhendajale ja klienditöö juhile. Rakenduse näitamise tagasiside oli väga positiivne. Tegevusjuhendaja tagasisidest selgus, et on kindel vajadus mobiilsete teavituste saatmisele. Seda kahjuks rakenduse praeguses iteratsioonis ei loodud, kuna teenus, mis saadab nutiseadmetesse tõukemärguandeid, on tasuline.

Nii klienditöö juht kui ka tegevusjuhendaja mainisid mõlemad, et näevad rakenduses palju väärtust ning sooviks ka rakendust klientidega testida. Implementeeritud funktsionaalsused töötasid nii, nagu esialgses intervjuus kirjeldati, ja kõik osad tundusid neile rakenduses vajalikud. Klienditööjuhi arvates oli parim loodud funktsionaalsus rakenduse sisene pildibaas, kus on võimalik klientidele määrata erinevaid piktogramme. Klienditööjuhi poolsest tagasisidest selgus ka, et lihtne klientide lisamine kiirendab oluliselt tööprotsessi. Nii on uute klientide hooldekodusse saabumisel kiire neid süsteemi sisestada. Olulise kohana tõi klienditööjuht välja selle, et ka temal oleks ülevaade klientide saadetud piktogrammidest. Tegevusjuhendaja nägi rakenduses olulist väärtust nii vestluskanali funktsionaalsuses kui ka klientide ülevaatliku listivaadet.

Negatiivse tagasisidena tõi klienditöö juht välja, et edasiarendusena oleks vaja luua kasutajaliides, mis on rohkem konfigureeritav vastavalt kliendile. Näiteks võimalus valida erinevate värvilahenduste vahel, aitamaks nägemispuudega kliente. Samuti oleks kasutajaliideses vaja kasutada rohkem värve, nii on kliendil parem piltide ja kasutajaliidese vahel seoseid luua.

## 4.5 Võimalikud edasiarendused

Bakalaureusetöö loomeprotsessi käigus ning arutades funktsionaalseid nõudeid intervjueeritavatega tekkis autoril mitmeid mõtteid rakenduse edasiarenduseks.

- Luua funktsionaalsed kasutajaliidese testid.
- Lisada võimalus määrata piktogrammidele suurusi.
- Lisada kategooriate funktsionaalsus.
- Lisada võimalus rakenduse kasutamiseks kujul, kus agregaadi juureks on olem Maja, sellega on tagatud võimalus rakendust tarnida ka kasutajatele, kes soovivad rakendust kasutada koduses keskkonnas.
- Lisada võimalus luua piktogrammidega päevakavad.
- Luua võimalus kasutada rakendust nii, et see muudaks seadme kõne genereerivaks seadmeks.
- Ühendada rakendus teavituste saatmise teenusega.
- Lisada võimalus kahepoolseks vestluseks kliendi ja tegevusjuhendaja vahel.
- Luua statistika vaade, lisades sinna teavitused, mis toovad esile vähe kasutust näinud piktogrammid.

- Rakendada kliendi piktogrammidele masinõppe algoritmi, mis ennustab kliendi saadetud piltide ja selle saatmise aega kasutades igale ajahetkele parimat pildid, mida klient võiks soovida saata.
- Luua süsteemile erinevad keele-paketid. Nii ei ole kasutajate arv piiratud keele järgi.
- Lisada funktsionaalsus, millega luuakse teatud aja tagant andmebaasidest koopia.
- Luua rakendusele soe koopia, mis rakendub, kui peamises rakenduses esineb katkestus. Nii on võimalik tagada rakenduse talitluspidevus .
- Krüpteerida sõnumivahetust hoidev andmebaas.
- Disainida kasutajaliides nii, et see oleks vastavuses WCAG [31] standarditele

## 4.6 Autori edasised plaanid

Kuna valdkond on tehnoloogilisest arengust maha jäänud, näeb autor võimalust panustada ka edaspidi oluliselt inimestesse, kes hetkel sellist vabadust tehnoloogiaga nautida ei saa.

Hetkel ei ole veel rakendus piisavalt valmis, et seda avaldada suuremasse keskkonda. Minimaalne rakendus, millega oleks võimalik toota reaalset väärtust nii hooldekodudes, kui ka peremajapidamistes vajaks järgnevaid funktsionaalsuseid, mis on järjestatud olulisuse järjekorras.

- Eraldatud sõnumite süsteem
- Krüpteeritud sõnumite andmebaas
- Disainida kasutajaliides, mis on kooskõlas WCAG [31] standarditega.
- Kliendi poolt saadetud piktogrammide kohta tagasiside saatmise võimalus
- Piktogrammide kategoriseerimise võimalus

Autoril on plaanis süsteemi arendamist jätkata ja järgmise etapina oleks vaja luua minimaalne rakendus, mille funktsionaalsused on kirjeldatud eespool. Minimaalset rakendust oleks võimalik jagada teenusena ka hooldekodudele. Nii saaks alustada tagasiside protsessiga, mille käigus kujuneks välja hooldekodude vajadustega kooskõlas süsteem, mis pakuks palju väärtust nii töötajatele kui ka klientidele.

# 5 Kokkuvõte

Bakalaureusetöö eesmärk oli luua rakendus, millega oleks võimalik tegevusjuhendajal ja puudega inimesel distantsilt vestelda. Peamiseks fookuseks oli vestluskanali loomine, kus puudega inimene saab saata piktogrammi tegevusjuhendajale.

Püsitatud eesmärgi saavutamiseks analüüsiti olemasolevaid rakendusi ja tehti intervjuu hooldekodu töötajatega. Olemasolevate rakenduste analüüsist selgus, et turul ei ole sellist lahendust, millega oleks võimalik luua vestluskanal puudega inimese ja tegevusjuhendaja vahel. Intervjuude ja vestluste käigus pakuti välja 16 funktsionaalset nõuet ja 10 mittefunktsionaalset nõuet. Klienditöö juhi poolt tulnud olulised nõuded olid: võimalus lisada uusi kliente ja tegevusjuhendajaid; võimalus valida klientidele erinevaid pilte, mis rakenduse baasis olemas on. Nõuetest ei suudetud kõiki funktsionaalseid nõudeid töö mahtu ja ajalist piirangut arvestades täita.

Töö nõudeid arvestades disainis autor esmalt äriloogika kihi, kasutades põhimõtteid, mis on kirjeldatud raamatus Domain-Driven Design [14]. Kui arhitektuur oli loodud, siis alustas autor rakenduse arendamisega. Arendamisel lähtuti Test Driven Development [30] printsiipidest.

Arendatud rakendusega täideti uurimistöös püstitatud eesmärk ja 10 funktsionaalset nõuet ning 9 mittefunktsionaalset nõuet. Tööd võib lugeda õnnestunuks, kuna demonstreerimisel saadud tagasiside oli positiivne ning töötajad nägid, et selline rakendus lihtsustaks oluliselt nende igapäevatööd.

# Kasutatud kirjandus


[1] Oxfordlearnersdictionaries, „pictogram noun - Definition, pictures, pronunciation and usage notes | Oxford Advanced Learner's Dictionary at OxfordLearnersDictionaries.com," [Võrgumaterjal]. Available: https://www.oxfordlearnersdictionaries.com/definition/english/pictogram#:~:text=%2F%CB%88p%C9%AAkt%C9%99%C9%A1r%C3%A6m%2F-,%2F%CB%88p%C9%AAkt%C9%99%C9%A1r%C3%A6m%2F,representing%20a%20word%20or%20phrase. [Kasutatud 24 04 2022].

[2] Sotsiaalkindlustusamet, „Puude raskusastme tuvastamise taotlemine," 06 01 2022. [Võrgumaterjal]. Available: https://www.eesti.ee/et/puudega-inimesed/puude-taotlemine/puude-raskusastme-tuvastamise-taotlemine. [Kasutatud 24 04 2022].

[3] A. Klaassen, A. Tiko, K. Mäe, M. Krais, M. Salumaa, P. Kokk, S. Agan, T. Arandi, U. Tõnisson ja Ü. Uusküla, Tegevusjuhendaja käsiraamat, Tallinn: Tallinna Raamatutrükikoja OÜ, 2010.

[4] Wikipedia, „Speech-generating device," [Võrgumaterjal]. Available: https://en.wikipedia.org/wiki/Speech-generating_device. [Kasutatud 24 04 2022].

[5] T. T. C. Solutions, „Talk Up! Pictograms Communicator," [Võrgumaterjal]. Available: https://play.google.com/store/apps/details?id=com.are.cb.pro. [Kasutatud 24 04 2022].

[6] J. LLC, „JABTalk," [Võrgumaterjal]. Available: https://play.google.com/store/apps/details?id=com.jabstone.jabtalk.basic. [Kasutatud 24 04 2022].

[7] E. Elram, „SymboTalk - AAC Talker," [Võrgumaterjal]. Available: https://play.google.com/store/apps/details?id=com.elelad.comboard. [Kasutatud 24 04 2022].

[8] BitProServices, „PicCom," [Võrgumaterjal]. Available: https://play.google.com/store/apps/details?id=com.bps.piccom. [Kasutatud 24 04 2022].

[9] S. Melo, „Cloud storage vs local storage: What is the right choice for you?," DataScope, 20 08 2021. [Võrgumaterjal]. Available: https://datascope.io/en/blog/cloud-storage-vs-local-storage-what-is-the-right-for-your-business. [Kasutatud 24 04 2022].

[10] Microsoft, „Visual studio code - code editing. redefined," Microsoft, 03 11 2021. [Võrgumaterjal]. Available: https://code.visualstudio.com/. [Kasutatud 24 04 2022].

[11] JetBrains, „Rider vs. Visual Studio - compare: Jetbrains Rider," JetBrains, [Võrgumaterjal]. Available: https://www.jetbrains.com/rider/compare/rider-vs-visual-studio/. [Kasutatud 24 04 2022].

[12] Docker, „Docker Overview,“ Docker, 22 04 2022. [Võrgumaterjal]. Available: https://docs.docker.com/get-started/overview/#:~:text=Docker%20allocates%20a%20read%2Dwrite,not%20specify%20any%20networking%20options. [Kasutatud 24 04 2022].

[13] A. Partil, „3 layered architecture,“ eCanarys, [Võrgumaterjal]. Available: https://www.ecanarys.com/Blogs/ArticleID/76/3-Layered-Architecture. [Kasutatud 24 04 2022].

[14] E. Evans, Domain-driven design: Tackling complexity in the heart of software., New Jersey, United States: Pearson Education (US), 2003.

[15] G. Alder, „Draw.Io,“ 2000. [Võrgumaterjal]. Available: https://app.diagrams.net/. [Kasutatud 24 04 2022].

[16] V. Paradigm, „What is Unified Modeling Language (UML)?,“ [Võrgumaterjal]. Available: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/. [Kasutatud 24 04 2022].

[17] A. Amadar ja S. Tandel, „Impact of Progressive Web Apps on Web App,“ *International Journal of Innovative Research in Science,,* kd. 7, nr 9, 09 2018.

[18] Vue.Js, „Introduction,“ Vue.Js, [Võrgumaterjal]. Available: https://vuejs.org/guide/introduction.html. [Kasutatud 24 04 2022].

[19] Vue, „Creating a project: Vue Cli,“ Vue CLI, [Võrgumaterjal]. Available: https://cli.vuejs.org/guide/creating-a-project.html#vue-create. [Kasutatud 24 04 2022].

[20] M. Fowler, „Bliki: Evansclassification,“ 14 12 2005. [Võrgumaterjal]. Available: https://martinfowler.com/bliki/EvansClassification.html. [Kasutatud 24 04 2022].

[21] M. Fowler, „Bliki: ValueObject,“ 14 11 2016. [Võrgumaterjal]. Available: https://martinfowler.com/bliki/ValueObject.html. [Kasutatud 24 04 2022].

[22] G. E. Krasner ja S. T. Pope, „A Description of the Model-View-Controller User,“ ParcPlace Systems, California.

[23] Okta, „Data Transfer Object DTO definition and usage,“ [Võrgumaterjal]. Available: https://www.okta.com/identity-101/dto/#:~:text=A%20data%20transfer%20object%20(DTO,for%20people%20with%20programming%20backgrounds. [Kasutatud 24 04 2022].

[24] M. Fowler, „Bliki: CQRS,” martinfowler.com,“ 14 07 2011. [Võrgumaterjal]. Available: https://martinfowler.com/bliki/CQRS.html. [Kasutatud 24 04 2022].

[25] Jbogard, „Jbogard/mediatr: Simple, unambitious mediator implementation in .NET,“ [Võrgumaterjal]. Available: https://github.com/jbogard/MediatR. [Kasutatud 24 04 2022].

[26] Wikipedia, „Document-oriented database,“ 16 12 2021. [Võrgumaterjal]. Available: https://en.wikipedia.org/wiki/Document-oriented_database. [Kasutatud 24 04 2022].

[27] M. Fowler, „Bliki: AggregateOrientedDatabase,“ 19 01 2012. [Võrgumaterjal]. Available: https://martinfowler.com/bliki/AggregateOrientedDatabase.html. [Kasutatud 24 04 2022].

[28] MinIO, „Minio for Amazon Elastic Kubernetes Service,“ [Võrgumaterjal]. Available: https://min.io/product/multicloud-elastic-kubernetes-service?utm_campaign=AWS%20EKS%201.0&utm_source=ppc&utm_medium=google&utm_term=aws-eks-marketplace&utm_content=aws-eks-

marketplace-0122&utm_term=aws%20s3&utm_campaign=MinIO+for+Google+Kubernetes+Engine+. [Kasutatud 24 04 2022].

[29] R. C. Martin, C. M. Feathers, T. R. Ottinger ja J. J. Langr, Clean code A handbook of agile software craftsmanship, Boston: Pearson Education, 2016.

[30] M. Fowler, „Bliki: TestDrivenDevelopment,“ 05 03 2005. [Võrgumaterjal]. Available: https://martinfowler.com/bliki/TestDrivenDevelopment.html. [Kasutatud 24 04 2022].

[31] A. G. W. Group, „WCAG 2 Overview,“ [Võrgumaterjal]. Available: https://www.w3.org/WAI/standards-guidelines/wcag/. [Kasutatud 17 5 2022].

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

Mina, Keith Roland Lepik

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose „Piktogramme kasutava puudega inimese ja tegevusjuhendaja vahelise kommunikatsiooni rakendus“, mille juhendaja on Liisa Jõgiste
   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.

17.05.2022

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 - Rakenduse kood

- Koodi repositoorium: https://github.com/KeithRolandLepik/SpeakToMe
- OrganizationController: https://github.com/KeithRolandLepik/SpeakToMe/blob/main/Api/Controllers/OrganizationController.cs
- HouseController: https://github.com/KeithRolandLepik/SpeakToMe/blob/main/Api/Controllers/HouseController.cs
- ClientController: https://github.com/KeithRolandLepik/SpeakToMe/blob/main/Api/Controllers/ClientController.cs