TARTU ÜLIKOOL

Arvutiteaduse instituut

Informaatika õppekava

Sergei Eensalu

Kasutaja haldamise tarkvara idufirmas *Rendin OÜ*

Bakalaureusetöö (9 EAP)

Juhendaja: Vambola Leping

Kaasjuhendaja: Alar Mäerand

Tartu 2020

Kasutaja haldamise tarkvara idufirmas *Rendin OÜ*

Lühikokkuvõte:

Käesoleva bakalaureusetöö koostamise käigus valmis Eesti eluruumi pikaajaliste üürilepingute haldamise kasutaja tarkvara. Rakendus aitab ettevõtte töötajatele hallata olemasolevaid kliendilepinguid, luua uusi ja lõpetada neid. Antud tarkvara annab võimaluse töödelda kliendi päringud ja hallata lepingud.

Võtmesõnad:

Rendin OÜ, ReactJS, Firebase, Redux, veebitarkvara ja haldussüsteem.

CERCS:

P175 Informaatika, süsteemiteooria

User management software for a start-up *Rendin OÜ*

Abstract:

In the course of the thesis, the user software for managing long-term leases of Estonian dwellings was completed. The application helps company employees to manage existing customer contracts, create new ones as well as terminate them. The software provides the ability to process customer inquiries and manage contracts/agreements.

Keywords:

Rendin OÜ, ReactJS, Firebase, Redux, web software and management system.

CERCS:

P175 Informatics, systems theory

Sisukord

Sissejuhatus

Tänasel päeval on Eestis väga keeruline leida eluruumi ja vormistada vajalikud dokumendid, sest selleks on vaja palju aega ja raha. Tavalises eluruumi otsimise ja allkirjastamise protsessis osalevad 3 osapoolt: üürikorteri otsija, üürileandja ja maakler. See protsess toob kaasa ka rahalised kulutused, sest sinna lisanduvad lepingutasu, maakleritasu, tagatisraha ja kolm kuud eluruumi renti ette. Eluruumi protsess võiks olla aga kordades lihtsam, odavam ja efektiivsem.

Eestis on kolm kõige populaarsemat keskkonda, mille kaudu otsitakse kinnisvara üürikuulutusi.
Eesti peamised kinnisvaraportaalid:

1. „KV.EE“ on turul aastast 1999, https://www.kv.ee/ .
2. „CITY24“ on turul aastast 2000, https://www.city24.ee/ .
3. „KINNISVARA24“ on turul aastast 2018, https://kinnisvara24.delfi.ee/ .

Rendin OÜ annab võimaluse sõlmida üürilepinguid turvaliselt ja efektiivselt, pakkudes kohustuste tagamise teenust, mis lepinguga sõlmides kaasnevad [1]. Antud teenusega kaasnev kindlustussertifikaat, mis väljastatakse nii üürileandjale kui üürnikule, tagab mõlema osapoole kindlustunde sõlmitud lepingu suhtes [1].

Rendin OÜ peamised tegevused on [1]:

- osapoolte taustakontroll;
- seadusega kooskõlastatud arusaadavad üürilepingud;
- lepingute vormistamine erinevates keeltes;
- kindlustus teenus nii üürnikule kui ka omanikule;
- pole vaja maksta tagatisraha ja sissemakset;
- veebipõhiselt üürilepingute allkirjastamine ja nende haldamine.

Antud töö eesmärk on kaasa aidata ja tuua turule kaasaegne kinnisvara otsimise lahendus ja ehitada veebirakendus, mida saavad kasutada Rendin OÜ töötajad kliendi lepingute loomiseks, haldamiseks ja lõpetamiseks.

Töö loomise etapid

- tutvuda erinevate kasutaja haldamise tarkvara süsteemidega;
- tutvuda Eesti kinnisvara portaalide iseteenindustega;
- koostada teoreetiline osa, mis kirjeldab tulevane veebirakendus;
- rakenduse loomine.

Töö struktuur

- sissejuhatus;
- iseteenindussüsteemide ülevaade;
- kinnisvaraportaalide kliendi iseteeninduste funktsionaalsuste ülevaade;
- ülevaade arhitektuurist ja kasutatud tehnoloogiast;
- rakenduse funktsionaalne ja mittefunktsionaalne kirjeldus;
- rakenduse loomise protsessi kirjeldus;
- kokkuvõte.

1. Kasutaja haldamise tarkvarasüsteem

Klienditugi nii väikestele kui ka suurettevõtetele on väga oluline. Usaldusväärse klienditoe süsteemi omamine annab ettevõtte tootemargile positiivse kuvandi ja näitab selgelt, et hoolitakse oma klientidest ning rahuldatakse kõik nende vajadused [2].

1.1 Kirjeldus

Kliendihalduse süsteem on andmebaas, mis sisaldab teavet kõigi ettevõtete klientide kohta, kes on sellega kunagi tehinguid teinud [3]. Lisaks viidatakse mõnikord ka kliendibaasile teavet võimalike ettevõtte klientide kohta [3]. Klienditeeninduse tarkvara võimaldab ettevõttel hallata, korraldada, reageerida ja vastata kõigile klienditeeninduse taotlustele. Programm aitab jälgida kõiki klientide taotlusi, kasutades ühe klõpsuga aruandeid, mõista reageerimise aegu, saada teateid vastamata e-kirjadest ja väga tähtsate klientide kiirete kirjade kohta ning lisada märkusi iga kliendi kohta [3].

Tarkvara klassifitseerimine [3]:

- **Üldandmete nimekiri kirjade postitamiseks.** Kliendibaas säilitab ainult kliendi nime ja e-posti/mobiilinumbri, et vajadusel saaks kliendiga ühendust võtta ja pakkumist esitada. Kliendibaasis on raske kontrollida, kas esitatud andmed on õiged, sest selleks on liiga vähe andmeid, mida saaks analüüsida. Näiteks, telefoni teel müügi tegemiseks on vaja läbi helistada inimesi, et saada neid kliendiks. Tavaliselt selle jaoks kasutatakse nii öelda telefoniraamatu loogikat.
- **Raamatupidamine.** Kliendibaas säilitab kliendi nime, tema rekvisiidid ja tellimuste ajaloo. Tavaliselt andmete õigsust kontrollitakse ja parandatakse tellimuse või tehingu tegemiseks. Näiteks, e-poest tellides tuleb kontrollida ja vajadusel parandada aadressi.
- **Laiendatud.** Sügavam andmebaas, kuhu sisestatakse kliendi andmed (nimi, kontaktnumber, email, isikukood, aadress). Kliendibaasi lisatakse kliendi eelistused,

jälgitakse kliendi rahulolu, määratakse kliendihaldur, kes aitab klienti ja vastutab kliendi tellimuste ja pöördumiste ajaloo eest.

1.2 Eesmärk ja vajadus

Ettevõtte kasvuga kasvab klientide arv ja mida rohkem kliente, seda raskem on iga klienti jälgida. Kui klientide arv on üle 100 inimese, siis on juba raske kasutada Outlooki, Excel tabelit või kalendrit, et jälgida iga endist, olemasolevat või tuleviku klienti [4]. Kliendi haldamise tarkvara võimaldab klientide andmete kaotamata hallata või muuta lepinguid, korrigeerida kliendi andmeid, jälgida arvete tasumist, näha kliendi pöördumisi ja pöördumistele reageerida.

Ühise kliendibaasi loomise eesmärgiks on klientide ja ettevõtete efektiivsete suhete tagamine ja andmete hoidmine [4]. Kliendibaasi loomisel on oluline organiseerida kliendihalduritele ligipääs, et ükskõik mis ajal ja kohas, oleks kliendihalduri jaoks kliendi andmed kättesaadavad [4].

Kasutaja haldamise süsteemi peamised eesmärgid [3-6]:

- **Väljavaadete analüüs.** Andmebaasi koostamine võimaldab turgu hinnata ettevõttele kuuluva levitamise protsendi ja klientide lojaalsuse järgi toote suhtes.
- **Andmete salvestamine.** Kliendibaas võimaldab ettevõtte juhil kindlustada ettevõtet klientide kaotamise vältimiseks, müügiesindaja või müügijuhi vallandamise korral. Sellise kokkuvõtte olemasolu võimaldab kiiresti ja tõhusalt tuua uusi töötajaid äritegevusse juurde.
- **Kliendi hindamine ja prognoosimine.** Kliendibaas võimaldab turusegmenti sügavamalt analüüsida. On võimalik saada infot oma klientide isiklike eelistuste kohta, hinnata nende ostukorvi, tellimuste sagedust ja hooajalisust. Selline analüüs võimaldab läbi viia turu-uuringuid ja teha prognoose klientide ja ettevõtte arengu kohta tervikuna.
- **Muude turu segmentide analüüs ettevõtte laiendamiseks.** Andmebaas kogub kliendi käitumist. Tänu sellise teabe kogumisele ja töötlemisele luuakse väljavaated seotud müügi teostamiseks või ettevõtte laiendamiseks.

- **Iga kliendi lähenemisviisi individualiseerimine.** Kliendibaas, mis segmendis uurib kliendi vajadusi ja valib igal üksikjuhul individuaalse lähenemisviis.

Kasutaja haldamise süsteemi peamised eelised [3-6]:

- Hoiab organiseeritust;
- Tuleviku, olemasolevate ja endiste klientide andmete/lepingute haldamine;
- Vajaliku korrigeerimise teostamist (sh lepingu lõpetamine/avamine):
    1. Kliendi pöördumise registreerimine;
    2. Kliendi laekumiste ja arve tasumise jälgimine;
    3. E-posti või lühisõnumi teavituse saatmine;
    4. Kampaaniapakkumised, lojaalsuse programm;
    5. Kliendi käitumise ja lepingu muutmise ajaloo jälgimine.
- Lihtne kliendi või lepingu otsimine;
- Klientide filtreerimine;
- Päevakava assistent/kalender;
- Kliendi tagasiside töötlemine.

1.3 Turvalisus

Klientide isikuandmete turvalisus ja nende andmebaasidest lekkimise teema on eriti aktuaalne, sest kliendi andmete turvalisus on väga kõrge prioriteediga. Andmete turvalisus peab olema tagatud nii tarkvaras (vältida tarkvara murdumist) kui ka töötaja poolt töötamise ajal. Andmete konfidentsiaalsuse eest vastutab ettevõte, kes andmeid hoiab. Klientide andmebaasid on sageli sattunud küberkurjategijate ja petturite küberrünnakute alla. Seda tehakse andmete hilisema müügi eesmärgil - paljud ettevõtted on valmis maksma suure summa üksikasjaliku teabe eest vajaliku turusegmendi klientide kohta. Samuti on teada teiste klientide kliendibaaside ostmise juhtumid - sarnane tava on eriti levinud kinnisvaraga kauplemise valdkonnas.

2. Eesti kinnisvara tarkvara müügiportaalide ülevaade

Eestis on kolm peamist kinnisvara otsimise ja müügiportaali. Antud bakalaureusetöö autor analüüsib eesti kinnisvara iseteeninduspportaale eraldi ja toob välja peamised iseteenindusbüroo funktsionaalsused.

2.1 City24

Kinnisvaraportaal City24 on veebikeskkond, mis on loodud kinnisvaramüüjate ning kinnisvara ostjate jaoks [7]. Tänapäeval City24 tegutseb nii Eestis, Lätis kui ka Leedus. City24.ee statistika ütleb, et City24 portaali külastab iga kuu umbes 188 000 unikaalset kasutajat [7].

City24 iseteeninduse funktsionaalsus:

- **Ülevaade.** Saab lisada uusi kuulutusi (korter, maja, majaosa, äripind, suvila, maa, garaaž);
- **Minu objektid.** Näitab kõik kasutaja aktiivseid ja mitteaktiivseid objekte;
- **Arved.** Arvete ajalugu ja info tasumise kohta;
- **Statistika.** Näitab kasutaja kuulutuse statistikat (kui palju kordi on kasutaja kuulutust vaadatud ning mitu korda salvestatud „lemmikutesse“);
- **Lemmikud.** Objektide salvestamise funktsioon, mis võimaldab huvitava objekti salvestada, et oleks paremini leitav;
- **Otsingud ja teavitused.** On salvestatud kasutaja otsingu tingimused, et järgmine kord oleks kiiremini sama otsingutulemused saadaval. Lisaks salvestatakse teavitused kasutaja otsingu kohta. See tähendab seda, et kasutaja saab teavituse, kus tekib uus kuulutus antud kasutaja otsingu tingimustega. On võimalik määrata kuhu teavitus saadetakse;
- **Viimati vaadatud.** Näitab kasutaja viimati vaadatud kuulutust;
- **Seaded.** Kasutaja haldamise leht:
    1. Kasutaja andmed (kasutajanimi, eesnimi, perekonnanimi, keel);

2. Turvalisus (senine parool, uus parool, salasõna);
3. Kontaktandmed (keel, telefon, e-post, uudiskirja nõusolek);
4. Arvete andmed (maksja nimi ja aadress).

2.2 Kinnisvara24

Kinnisvaraportaal Kinnisvara24 on veebikeskkond, mis annab kasutajale infot Eesti kinnisvara kohta [8]. Selle kinnisvara portaali eesmärk on olla Eesti turul mugavam ja parim kinnisvaraportaal [8].

Kinnisvara24 iseteeninduse funktsionaalsus:

- **Minu kuulutused.** Antud lehel on kõik kasutaja aktiivsed ja mitteaktiivsed kuulutused;
- **Lisa kuulutus.** Saab lisada uusi kuulutusi (korter, maja, majaosa, äripind, suvila, maatükk, garaaž);
- **Salvestatud otsingud.** On salvestatud kasutaja otsingu tingimused, et järgmine kord oleks kiiremini sama otsingu tulemused saada;
- **Lemmikud.** Objektide salvestamise funktsioon, mis võimaldab huvitava objekti salvestada, et oleks paremini leitav;
- **Seaded.** Kasutaja haldamise leht:
    1. Kasutaja andmed (kasutajanimi, eesnimi, perekonnanimi, pilt);
    2. Turvalisus (senine parool, uus parool);
    3. Kontaktandmed (telefon, e-post, uudiskirja nõusolek).

2.3 KV.EE

KV.EE tutvustab ennast oma kodulehel nii, et KV.EE on esimene kinnisvaraportaal Eestis, mis oli loodud aastas 1999 [9]. Tänapäeval KV.EE tegutseb üle 20 aastat ja on peamine kinnisvarakuulutuste portaal Eestis [9]. KV.EE portaali peamiseks eesmärgiks on jagada otsijatele parimaid kuulutusi vastavalt hetkel turul olevatele trendidele [9]. Lisaks sellele,

antud kinnisvara veebilehekülg, aitab kinnisvarabüroodel ja kinnisvara omavatel eraisikutel avaldada lihtsalt kuulutusi ning neid hallata mugavas keskkonnas [9].

KV.EE iseteeninduse funktsionaalsus:

- **Minu kuulutused.** Antud lehel on kõik kasutaja aktiivsed ja mitteaktiivsed kuulutused;
- **Lisa kuulutus.** Saab lisada uusi kuulutusi („soovin müüa“, „annan üürile“, „lühiajaline üür“);
- **Seaded.** Kasutaja haldamise leht:
    1. Kasutaja andmed (kasutajanimi, eesnimi, perekonnanimi);
    2. Turvalisus (senine parool, uus parool);
    3. Kontaktandmed (telefon, e-post).
- **Minu arved.** Kliendil on võimalik jälgida oma olemasolevaid ja alles hiljuti esitatud arveid. Esitatud arved on seotud kliendi tasuliste teenustega, näiteks kuulutuse lisamise tasu;
- **Lemmikud.** Objektide salvestamise funktsioon, mis võimaldab huvitava objekti salvestada, et oleks paremini leitav;
- **Salvestatud otsingud/e-agendid.** On salvestatud kasutaja otsingu tingimused, et järgmine kord oleks kiiremini sama otsingu tulemused saada. Lisaks salvestatakse teavitused kasutaja otsingu kohta. See tähendab seda, et kasutaja saab teavituse, kus tekib uus kuulutus antud kasutaja otsingu tingimustega. On võimalik määrata kuhu teavitus saadetakse;
- **Juhend „Kuidas lisada kuulutus“ ja hinnakiri.**

3. Uue tarkvara nõuded

Käesolev peatükk kirjeldab autori ja kliendi koostööd ning valminud nõudeid uuele veebitarkvarale, mis sisaldavad nii funktsionaalseid- kui ka mittefunktsionaalseid nõudeid.

Kliendiga suhtlemine toimus erinevatel viisidel. Esimesel kohtumisel kliendiga tõime välja uue tarkvara soovitava funktsionaalsuse, tarkvara struktuuri ja tehnoloogiad ning planeerisime ajaliselt arendusprotsessi. Ettevõtte peakontoris toimusid esmaspäeviti esitluse koosolekud, kus autor näitas tarkvara hetkeseisu, ja toimusid tarkvara suurte osade läbirääkimised. Kliendil ja autoril oli võimalus esitada küsimusi, anda tagasisidet ja teha ettepanekuid olemasolevale või loodavale tarkvara osale. Jooksvaid küsimusi arutati kasutades Slack'i ja telefoni teel.

Tarkvara arendamine toimus nullist, sest kliendil puudus varasem lahendus. Lisaks puudus ka prototüüp, seega autor sai ise välja mõelda veebitarkvara kujunduse.

3.1 Funktsionaalsed nõuded

Veebirakendusel kehtivad üldised veebirakenduse nõuded ja tavad. Lisatud on peamine- ja lisafunktsionaalsus, mis on kliendi ja autori poolt paika pandud.

Üldised nõuded:

1. Kasutaja saab keskkonda sisse ja välja logida;
2. Igal kasutajal on oma õigused ja kõik õigused on erinevad;
3. Kasutaja saab navigeerida kliendiandmete ja lepingute detailse vaade, otsingu ja ülesannete lehe vahel;
4. Andmed peavad olema kuvatud niimoodi, et saaks vajadusel mitte olulised andmed peita ja töötada ainult oluliste andmetega. Ruumi optimeerimine.

Sisse logimata kasutaja funktsionaalsed nõuded:

- Esileht:

    1. Kasutaja saab sisse logida keskkonda;
    2. Kasutajale kuvatakse lehekülje veebikeskkonna info;
    3. Kasutaja saab ettevõtte märgile vajutades liikuda sisse logimata kasutaja esilehele;
    4. Kasutaja saab vastava veateade, kui sisse logitakse vigaste andmetega;
    5. Kasutaja sisse logides suunatakse sisse logitud kasutaja esilehele.

Sisse logitud kasutaja funktsionaalsed nõuded:

- Navigatsiooniriba

    1. Kasutaja saab liikuda kliendi andmete ja lepingu otsimise lehele;
    2. Kasutaja saab liikuda ülesannete lehele;
    3. Kasutaja saab välja logida.

- Kliendi otsimise leht:

    1. Kasutaja saab otsida klienti.
        1.1 Otsimise võtmed: isikukood, mobiilinumber, kliendinumber, ettevõtte registrikood või e-post.
    2. Kasutaja näeb loetelu kõikidest leitud klientidest, kui leitud rohkem kui üks klient.
        2.1 On näha peamised andmed: ees- ja perekonnanimi, isikukood, e-posti aadress.
        2.2 Kasutaja saab valida loetelust õige kliendi ja avatakse kliendi andmete detailne vaade.
    3. Kasutaja suunatakse kliendi andmete lehele, kui on leitud ainult üks klient.
    4. Kasutaja saab otsida lepingut.
        4.1 Otsimise võtmed: lepingu number, andmebaasi id number.
        4.2 Lepingu number on alati unikaalne, seega kui lepingu number kasutades leitakse vaste, siis kohe avatakse lepingu vaatamise leht.

- Kliendi andmete leht:

1. Kasutaja näeb avatud kliendi andmeid ja lepinguid.
   1.1 Kliendi andmetest kuvatakse ees- ja perekonnanimi, e-posti aadress, isikukood, kliendinumber, kliendi loomise kuupäev, mobiilinumber, krediidi taustakontroll, postiaadress, ettevõtte nimi ja registrikood, kui tegu on ettevõttega.
2. Kasutaja saab kliendi andmeid muuta.
   2.1 Kasutajale avatakse kliendi andmete muutmise võimalus, kui tema vajutab andmete muutmise nuppu.
   2.2 Kasutaja saab salvestada andmed või loobuda salvestamisest, vajutades vastavat nuppu.
   2.3 Andmete salvestamise ajal toimub andmete kontroll.
3. Kasutaja näeb kõiki kliendi lepinguid.
   3.1 Lepingud kuvatakse tabelina.
   3.2 Iga lepingu peal on näha lepingu number, postiaadress, lepingu staatus ja kliendi roll.
   3.3 Lepingud sorteeritakse kliendi rolli järgi.
4. Kasutaja saab vajutada lepingu peale ja avada lepingu detailse vaate.
5. Kasutaja saab kliendi andmete osa peita, et vabastada ruumi veebilehel.
6. Kasutaja saab lepingu listi osa peita, et vabastada ruumi veebilehel.

- Lepingu leht:

1. Kasutaja näeb lepingu andmeid.
   1.1 Lepingu andmetest kuvatakse üüriobjekti, arvelduse, rendiperioodi, eritingimuste, kaaselanike ja osapoolte andmed ning dokumendid.
   1.2 Kasutaja saab peita plokid (alamkategooriaid).
2. Kasutaja saab lepingu andmeid muuta.
   2.1 Kasutajal avatakse lepingu andmete muutmise võimalus, kui isik vajutab andmete muutmise nuppu.

2.2 Kasutaja saab salvestada andmeid või loobuda salvestamisest, vajutades vastavat nuppu.

2.3 Andmete salvestamise ajal toimub andmete kontroll.

3. Kasutaja saab genereerida ERGO jaoks vajaliku PDF-faili raporti vajutades vastavat nuppu.

3.1 Raportis kuvatakse lepingu osapoolte ja kinnisvaraobjekti andmed, ERGO sertifikaadid ja kindlustuse summad.

4. Kasutaja saab lisada lepingusse vajaliku töö jaoks informatsiooni või ülesannet.

4.1 Paremal pool asub lahter, kuhu kasutaja saab kirjutada informatsiooni ja seda salvestada.

4.2 Kasutaja saab määrata, kas antud informatsioon on lihtsalt "jooksev info" või see on veel pooleli olev ülesanne.

4.3 Kui see on pooleliolev ülesanne, siis see kuvatakse üleval esimesena ja on nähtav. Ülesande saab lõpetada vajutades vastavat nuppu.

4.4 Lõpetatud ülesanne ei ole rohkem aktiivne, vaid on tavaline informatsioon.

4.5 Iga informatsiooni ja ülesande kohta on näha loomise ja lõpetamise kuupäev, staatus, looja e-posti aadress ja kirjeldus.

5. Kasutaja saab lisada dokumente.

5.1 Vajutades vastavat nuppu, kasutaja saab valida oma kõvakettalt faili ja lisada selle.

5.2 Kasutaja näeb rohelist info, et fail oli lisatud.

5.3 Kasutaja saab lisada ainult PDF, JPD, PNG ja ASICE tüüpi faile.

6. Kasutaja saab avada lepingu dokumente.

6.1 Kasutajale kuvatakse kõik lepingute dokumendid tabelina.

6.2 Dokumentide tabelis on näha faili nimi, faili väike pilt ja faili tüüp.

6.3 Vajutades dokumendi peale avatakse eraldi aknas vastav fail.

7. Kasutaja saab sulgeda kõik lepingud ja kliendi andmed ning tühjendada otsingut.

7.1 Menüüs asub vastav nupp, mis tühjendab otsing ja sulgeb avatud andmed.

- Ülesannete leht

1. Kasutaja näeb aktiivsete ülesannete nimekirja.
   1.1 Iga ülesanne kuvatakse eraldi kastina, mida saab peita või avada.
   1.2 Iga kasti peal on ülesande kirjeldus.
2. Kasutaja saab vaadata ülesande detailset vaadet, vajutades vastava ülesande peale.
3. Ülesandes on kirjas millal ja kes selle ülesande tegi, kirjeldus, lepingu number ja selle lepingu osapooled.
   3.1 Kasutaja saab liikuda lepingu detailse vaatele, vajutades vastavat nuppu.
   3.2 Kasutaja saab liikuda lepingu osapoole detailse vaate peale, vajutades vastavat nuppu.
   3.3 Kasutaja saab ülesadne sulgeda vajutades ‘tehtud’ nupu peale.

3.2 Mittefunktsionaalsed nõuded

1. Veebileht on ehitatud loogikaga *Simple web page*, mis tähendab seda, et mallid peavad olema minimalistlikud ja mida vähem alamlehe seda parem.
2. Veebileht ei pea olema mobiilisõbralik. Piisab veebisõbralikust keskkonnast.
3. Veebileht peab toetama ainult inglise keelt.
4. Veebilehe kujundus peab vastama headele tavadele.
5. Veebisait peab toetama vähemalt kolme neljast levinumast veebibrauserist.

4. Kasutatud tehnoloogiad ja keeled

Käesolev peatükk kirjeldab autori ja kliendi poolt valitud ja kasutatud tehnoloogiaid, programmeerimiskeeli ja annab teada miks antud otsus on tehtud.

Kliendil puudusid rangeid reegleid tehnoloogia valiku osas, seega autor valis tehnoloogiad ise. Klient hoidis oma ettevõtte andmed andmebaasis Firebase.

Veebirakenduse loomisel on kasutatud järgmisi keeli ja tehnoloogiad:

1. JS (JavaScript);
2. NoSQL (not only SQL);
3. CSS (Cascading Style Sheets).

Veebirakenduse loomisel on kasutatud järgmisi raamistikke:

1. ReactJS;
2. Bootstrap 4.

Keelte, raamistikkude ja tehnoloogiate valik lähtus autori varasematest kogemustest ning kokkulepitud kliendiga tehnoloogiatest.

4.1 JS (JavaScript)

JavaScript on skriptimise ja objektorienteeritud programmeerimiskeel, mis sisaldab palju funktsioone peaasjalikult veebilehtede skriptimiseks [10]. See on keel, mis võimaldab veebilehel kasutada keerukaid asju - iga kord, kui veebilehel juhtub mingi enamat kui lihtsalt staatiline kuvamine - võimaldab luua dünaamiliselt värskendatud sisu, juhib multimeediumit, animeerib pilte [10].

Selle tarkvara tegemise jaoks oli valitud JavaScript, sest seda keelt toetatakse kõigis operatsioonisüsteemides, igat tüüpi brauserites ning lauaarvutites ja mobiilseadmetes. Lisaks, antud keel on tuttav ettevõtte arendajatele, kes vajaduse korral saavad autoriga konsulteerida, aidata ja/või tulevikus jätkata, selle lõputöö raames loodud tarkvara arendamisega.

Enamus tööst, antud lõputöö raames, oli kirjutatud kasutades JavaScript'i.

4.2 NoSQL

NoSQL (eng. not only SQL) on lähenemisviis andmebaaside kujundamisele, mis mahutab väga mitmesuguseid andmemudeleid ja mis on mitterelatsiooniline alternatiiv traditsioonilistele relatsioonilistele andmebaasidele [11]. Suur erinevus relatsioonilistest andmebaasidest on see, et mitterelatsiooniline NoSQL andmebaas ei vasta rangelt ette määratud andmemudelile [11].

Antud tehnoloogia oli valitud kliendi poolt. Klient otsustas, et soovib kasutada Firebase infrastruktuuri, seega kliendi jaoks ei olnud teisi võimalusi kui valida NoSQL andmebaasi tehnoloogia. Lisaks võimaldab see kiiresti edasi liikuda, sest on võimalik andmemudelit kiiresti ja lihtsalt täiendada ja tulevikus lihtsalt kliendi süsteemi skaleerida.

4.3 CSS

CSS on veebilehtede kujundamiseks keel [12]. Kõige rohkem kasutatakse HTML- ja XHTML- lehtede malli loomisel [12].

Antud keel oli valitud tarkvara veebilehe kujundamise loomise abikeele jaoks. Antud projektis CSS kasutus on väga väike, aga oli mõnes kohas kasutatud värvide, stiilide ja üksikute plokkide asukoha määramise jaoks.

Nii autor kui ka kliendi poolsed arendajad olid varasemalt tutvunud CCS keelega, seega ei tekkinud osapoolte vahel arutelu ja antud keele valiku otsus oli tehtud kiiresti.

4.4 ReactJS

ReactJS on JavaScript'i raamistik kasutajaliideste loomiseks [13]. See on veebirakenduste vaate kiht. React võimaldab arendajatel luua suuremahulisi veebirakendusi, mis kasutavad andmeid, kujundusi ja muutuvad ajas, ilma et veebilehte uuesti laadima peaks [13].

Autori poolt oli pakutud kaks veebiraamistikku varianti: AngularJS ja ReactJS. Autor on kokku puutunud mõlema raamistikutega ja autori kogemus laseb kasutada mõlemaid. Kliendi poolt ei tehtud ettepanekut seoses raamistiku valimisega. Tänapäeval ReactJS on üks kõige populaarsematest veebiraamistikest maailmas, seega lõplikuks valikuks osutus ReactJS. Lisaks, antud raamistik on tuttav ettevõtte ühele arendajale, kes vajaduse korral saab autoriga konsulteerida, aidata ja/või tulevikus jätkata, selle lõputöö raames loodud tarkvara arendamisega.

4.5 Bootstrap 4

Bootstrap on CSS-i raamistik, mida kasutatakse enamuste veebiarenduste kujunduste loomisel [14]. Bootstrap sisaldab CSS- ja JavaScript'il põhinevaid kujundusmalle tüpograafia, vormide, nuppude, navigeerimise ja muude liidese komponentide jaoks [14].

Kliendi soov oli osta uue tarkvara jaoks sobiva Bootstrap mall, mis oleks mugav ja ilus tabelite tegemise jaoks. Autori poolt oli pakutud selline variant, et alguses kasutada tasuta Bootstrap malli varianti, kus on põhi kujundus ja värvid olemas ning kui tarkvara projekt on lõpusirgel, siis osta uus kujundus, asendades põhi mall ning rakendada tarkvarasse.

5. Uue veebilehe arendamise protsess

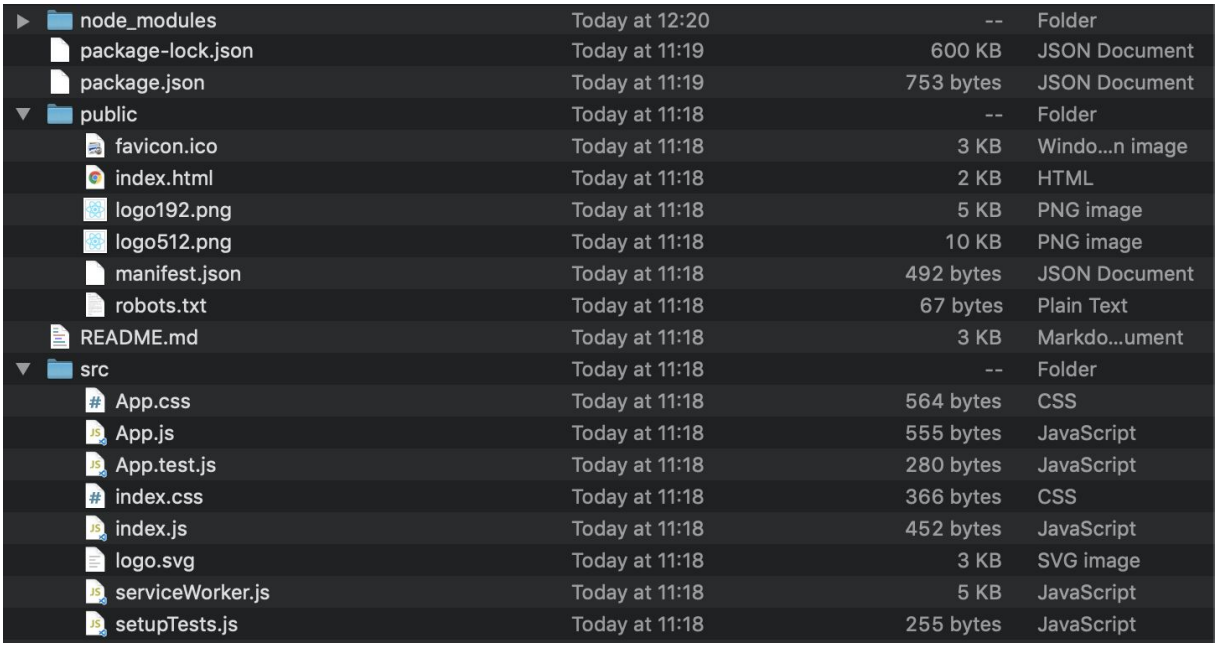
Käesolev peatükk kirjeldab uue veebilehe arendamise protsessi. Peatükis kirjeldatakse ReactJS projekti loomist ja püsti panemist ning erinevate tehnoloogiate integreerimist. Näiteks: andmebaasi ühenduse loomine, Redux'i integreerimine.

5.1 ReactJS projekti loomine

React projekt on deklaratiivne, tõhus ja paindlik JavaScript'i teek kasutajaliideste loomiseks [15]. See võimaldab koostada keerulisi kasutajaliideseid väikestest ja eraldatud koodi tükkidest, mida nimetatakse komponentideks [15]. Projekti loomisel esmaselt installeeriti vajalikud sõltuvused ja genereeriti projekti esialgne struktuur [16].

```
npx create-react-app rendin-admin
```

Joonis 1. Esialgse projekti loomine käsurealt.



Joonis 2. Automaatselt genereeritud failipuu.

Joonisel 2 kujutatud kataloogid ja failid[16]:

1. Väline juurkataloog *rendin-admin/* sisaldab endas kõiki projekti alamkatalooge ja faile;
2. Sisene kataloog *node_modules*/ sisaldab endas kõiki sõltuvusi, mida meie React rakendus nõuab. Antud kataloog sisaldab peaaegu 40 000 faili ja selle;
3. Sisene fail *package.json* sisaldab endas projekti spetsifikatsiooni: projekti versioon, sõltuvuste versioonid, projektitöö skriptid ja muud konfiguratsioonid;
4. Sisene kataloog *public*/ sisaldab kõiki staatilisi faile. Antud failide nimed peavad olema samad, kui tehakse tootepaigaldus;
5. Sisene fail *README.md* sisaldab dokumentatsioon kõikidest projekti failidest ja kataloogidest. Antud fail sisendab juhendit uutele meeskonnaliikmetele arengukeskkonna seadistamise ja alustamise jaoks;
6. Sisene *src*/ kataloog sisaldab endas kõik projekti komponendid. Antud kataloog on üks kõige olulisematest kataloogidest ja kasutatakse projekti loomise jaoks;
7. File *App.js* on *ReactJS* projekti kõikide komponentide peamine fail;

5.2 React JS projekti käivitamine

Projekti käivitamise jaoks on vaja käivitada käsurealt käsk, asudes samal ajal projekti juurkataloogis *rendin-admin/* .

```
npm start
```

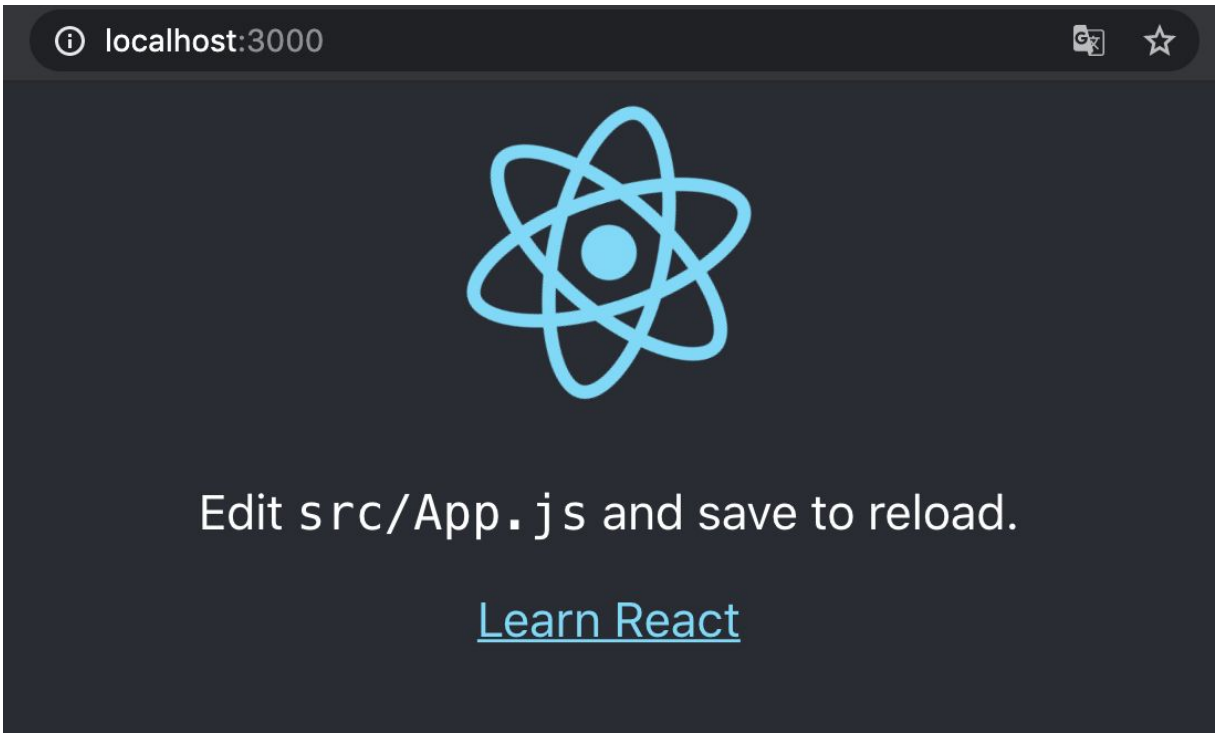
Joonis 3. React JS projekti käivitamine.

Projekti käivitamisel on näha, et kompileerimine õnnestus ja lokaalse projekti asukoht.

```
Compiled successfully!

You can now view rendin-admin in the browser.

  Local:            http://localhost:3000/
  On Your Network:  http://192.168.0.20:3000/

Note that the development build is not optimized.
To create a production build, use npm run build.
```

Joonis 4. React JS projekti edukas käivitamise käsurea tulemus.

Eduka kompileerimise korral avatakse automaatselt veebibrauser koos lingiga. Käivitatud projekt asub aadressil http://localhost:3000/



Joonis 5. Veebibrauseri tulemus.

Muutes koodi, laetakse leht paari sekundi jooksul automaatselt uuesti. Käsureas on näha ehitamise, kompileerimise vigu ja hoiatusi.

5.3 React JS projekti Redux'i lisamine

Redux on JavaScript'i raamistik, mis aitab hallata rakenduse andmete olekut [17].

Redux installeerimise jaoks on vaja käivitada käsurealt käsk, asudes samal ajal projekti juurkataloogis *rendin-admin/* .

```
npm install react-redux
```

Joonis 6. Redux installeerimise käsk.

Ühendame andmete hoidmise olekut (<*Provider store={store}* />) koos projekti <*App* /> rakendusega. <*Provider* /> teeb Redux'i oleku kättesaadavaks kõigile rakenduse komponentidele.

```
export const store = createStore(
    allReducers,
    enhancer: window.__REDUX_DEVTOOLS_EXTENSION__ && window.__REDUX_DEVTOOLS_EXTENSION__(),
);

ReactDOM.render(
    <Provider store={store}>
        <App/>
    </Provider>, document.getElementById( elementId: 'root'));


serviceWorker.unregister();
```

Joonis 7. Ühendatud andmete hoidmise olek koos peamise projekti klassiga.

Toimingud (eng. actions) on ainukesed funktsioonid, mis edastavad informarmatsiooni rakendusest olekule [18]. Toimingud kirjeldavad ainult mis juhtub, kuid ei kirjelda, kuidas rakenduse olek muutub [18].

```
export const CLEAR_TASKS_FROM_LIST = () => {
    return {
        type: 'CLEAR_TASKS_FROM_LIST'
    }
};

export const DELETE_TASK_FROM_LIST = (id) => {
    return {
        type: 'DELETE_TASK_FROM_LIST',
        id
    }
};

export const ADD_SEACRH = (search) => {
    return {
        type: 'ADD_SEACRH',
        search
    }
};

export const CLEAN_SEARCH = () => {
    return {
        type: 'CLEAN_SEARCH'
    }
};
```

Joonis 8. Toimingute funktsioonide näidis .

Reduktorid (eng. reducers) määravad, kuidas rakenduse olek muutub vastavalt saadetud toimingutele [19].

```
const tasksReducer = (state :*[] = [], action) => {
    switch (action.type) {
        case 'ADD_TASKS_TO_LIST':
            return action.tasks;
        case 'CLEAR_TASKS_FROM_LIST':
            return [];
        case 'DELETE_TASK_FROM_LIST':
            return state.filter(item => item.id !== action.id);
        default:
            return state;
    }
};

export default tasksReducer;
```

Joonis 9. Oleku muutmise näidis.

Redux ja ReactJS komponendi ühendamine omavahel toimub kasutades funktsiooni connect() [20]. Funktsioon seob komponenti andmehoidla jaoks vajalikke andmeid ja funktsioone, mida kasutatakse toimingute hoidlasse saatmiseks [20]. Funktsioon mapStateToProps jälgib komponenti Redux oleku värskendusi. See tähendab, et mapStateToProps kutsutakse iga

kord, kui Redux olek muutub [20]. Ühtegi toimingut ei ole võimalik lihtsalt saadetise (eng. dispatch) kaudu välja kutsuda, seetõttu mapDispatchToProps kaudu saab saata funktsioon rekvisiitidega tagasi helistamise(eng. callback) teel [20].

```
function mapStateToProps(state) {
    return {
        profile: state.selectedProfile,
        foundProfilesList: state.foundProfilesList,
        openContractsList: state.openContractsList,
        agreementItemsList: state.agreementItemsList,
        tasksList: state.tasksList,
        search: state.search
    }
}

function matchDispatchToProps(dispatch) {
    return bindActionCreators( actionCreators: {
        ADD_PROFILE: ADD_PROFILE_DATA,
        CLEAR_PROFILE_DATA: CLEAR_PROFILE_DATA,
        ADD_PROFILES_LIST: ADD_PROFILES_LIST,
        CLEAR_PROFILES_LIST: CLEAR_PROFILES_LIST,
        CLOSE_OPEN_AGREEMENTS: clearAllAgreementFromOpenList,
        ADD_AGREEMENTS_TO_TENANT_LIST: ADD_AGREEMENTS_TO_TENANT_LIST,
        ADD_AGREEMENTS_TO_LANDLORD_LIST: ADD_AGREEMENTS_TO_LANDLORD_LIST,
        ADD_AGREEMENTS_TO_CREATOR_LIST: ADD_AGREEMENTS_TO_CREATOR_LIST,
        CLEAN_ALL_AGREEMENTS_FROM_AGREEMENTS_LIST: CLEAN_ALL_AGREEMENTS_FROM_AGREEMENTS_LIST,
        CLEAR_TASKS_FROM_LIST: CLEAR_TASKS_FROM_LIST,
        ADD_SEACRH: ADD_SEACRH,
        CLEAN_SEARCH: CLEAN_SEARCH,
        CLOSE_TASK_PAGE: CLOSE_TASK_PAGE,
        ADD_AGREEMENT_TO_OPEN_LIST: ADD_AGREEMENT_TO_OPEN_LIST,
        REMOVE_CLIENTS_FROM_LIST: REMOVE_CLIENTS_FROM_LIST,
        CLOSE_ALL_CLIENTS_PAGE: CLOSE_ALL_CLIENTS_PAGE
    }, dispatch)
}


export default connect(mapStateToProps, matchDispatchToProps)(Search);
```

Joonis 10. ReactJS komponent ja Redux ühenduse näidis.

5.4 Firebase andmebaasi ühenduse loomine

Firebase pakub lihtsaid tööriistu ja infrastruktuuri, mis aitavad kaasa rakenduse arendamisele [21]. Firebase Security Rules tehnoloogia aitab kaitsta andmebaasi andmed [21]. Tänu antud tehnoloogiale on võimalik seadistada, kes saab mingeid tabeli andmed luua, lugeda ja kirjutada [21].

```
match /files/{document=**} {
  allow write:  if  isAdmin();
  allow read:   if  isAdmin();
  allow create: if  isAdmin();
}
```

Joonis 11. Failide kollektsioon tabeli loomise, kirjutamise ja lugemise õigused.

Firebase mooduli paigaldamise jaoks on vaja käivitada käsurealt käsk , asudes samal ajal projekti juurkataloogis *rendin-admin/* [22].

```
npm init
npm install --save firebase
```

Joonis 12. Firebase käsud mooduli paigaldamise jaoks.

Andmebaasi juurdepääsu saame tänu defineeritud Rendin Firebase konfiguratsioonile ja **firebase.initializeApp()** funktsioonile [22]. Enne andmebaasi konfiguratsiooni toimub kontroll, kas kasutatakse arendus- või toodangukeskkonda ja vastava tulemusele defineeritakse andmebaasi võtmed.

```
const developmentConfig = {
    apiKey: 'XXXXXXXXXXXXXXXXXXXXX-XXXXXXXXXXXXXX',   // TEST DATABASE
    authDomain: 'rendin.firebaseio.com',
    databaseURL: 'https://rendin.firebaseio.com',
    projectId: 'rendin',
    storageBucket: 'gs://rendin.appspot.com',
    messagingSenderId: 'Rendin',
};

let database;

if (isDevelopmentMode) {
     database = firebase.initializeApp(developmentConfig);
} else {
    database = firebase.initializeApp(productionConfig);
}
```

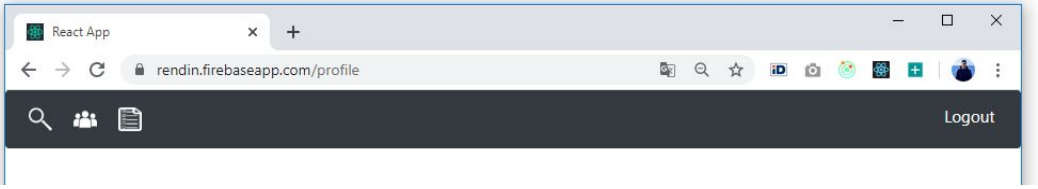
Joonis 13. Firebase andmebaas ja ReactJS ühenduse loomine.

6. Tulemuse ülevaade ja analüüs

Käesolev peatükk kirjeldab uue veebitarkvara saadud tulemust ja vastavust kliendi nõuetele. Peatükis kirjeldatakse loodud funktsionaalsust, kliendi võimalikku käitumist, loodud tulemuse skeemi ja projekti koodi osasid.

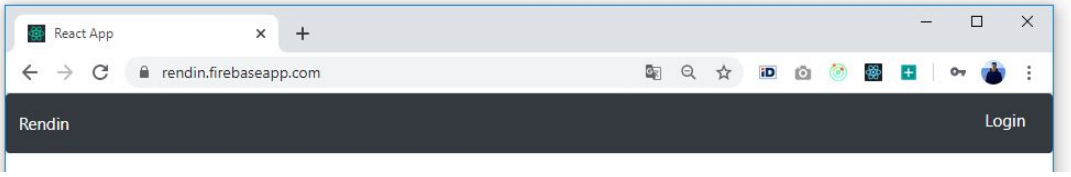
6.1 Päis

Loodud üks universaalne veebilehe päis, mis on kasutusel igal veebi alamlehel. Veebilehe päise funktsionaalsus muutub vastavalt sellele, kas kasutaja on sisse logitud või mitte. Sisse logitud kasutaja näeb päises otsingut, kliendi baasi ja ülesannete nuppe, mille kaudu saab liikuda vastava alamlehele või välja logida süsteemist.



Joonis 14. Uue tarkvara päis, kui kasutaja on sisse logitud.

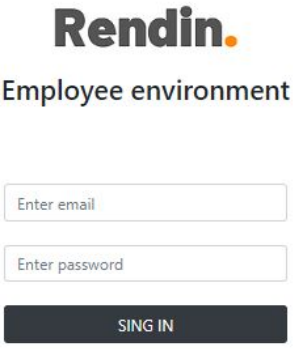
Sisse logimata kasutaja näeb ainult firma märki ja sisselogimise nuppu.



Joonis 15. Uue tarkvara päis sisse logimata kasutaja korral.

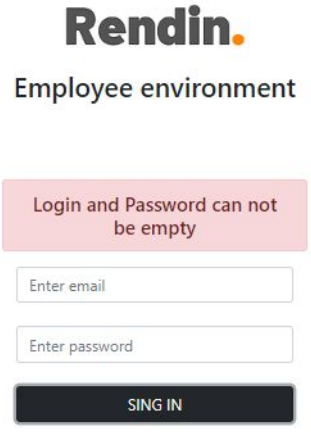
6.2 Sisselogimise leht

Kasutaja saab sisse logida, kasutades selleks kasutajatunnust (e-post) ja salasõna.



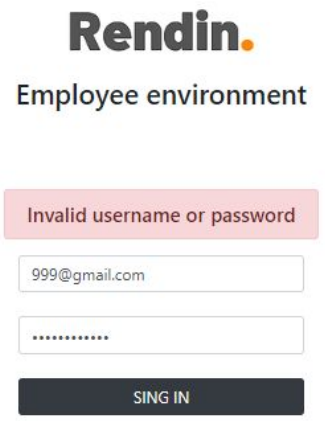
Joonis 16. Veebikeskkonda sisselogimise leht.

Kasutaja saab vastavad veateated, kui sisselogimine mingil põhjusel ei õnnestunud.



Joonis 17. Sisselogimise veebikeskkonna leht koos vastava veateadega.

Turvalisuse tagamiseks on kasutatud üldist veateadet, seega kui kasutaja paneb õige kasutajatunnus ja vale parooli, siis ta saab üldise veateate, et on sisestatud “vale kasutajanimi või parool”. Turvalisuse huvides süsteem ei anna teada, mis täpsemalt on valesti sisestatud, seega see aitab vältida kõrvalistel isikutel kontole ligipääsu saada.



Joonis 18. Sisselogimise veebikeskkonna leht koos üldise veateadega.

Süsteemi pole võimalik registreerida ja salasõna unustamise korral parooli taastada. Kasutajatunnuste ja paroolide saamiseks või parooli unustamise korral, saab luua konto või anda uue parooli ainult vastav inimene firmas.

Autentimise jaoks kasutatakse Firebase Authentication programmi, mis aitab autentida kasutajaid nende e-posti aadressite ja salasõnade abil.

```
await firebase.auth()
    .signInWithEmailAndPassword(this.state.email, this.state.password)
    .catch(() => {
        this.setState({
            error: Translation.EMPLOYEE_LOGINING_BASIC_ERROR,
            hideAlert: false
        });
    });
```

Joonis 19. Firebase Authentication integreerimise ülevaade.

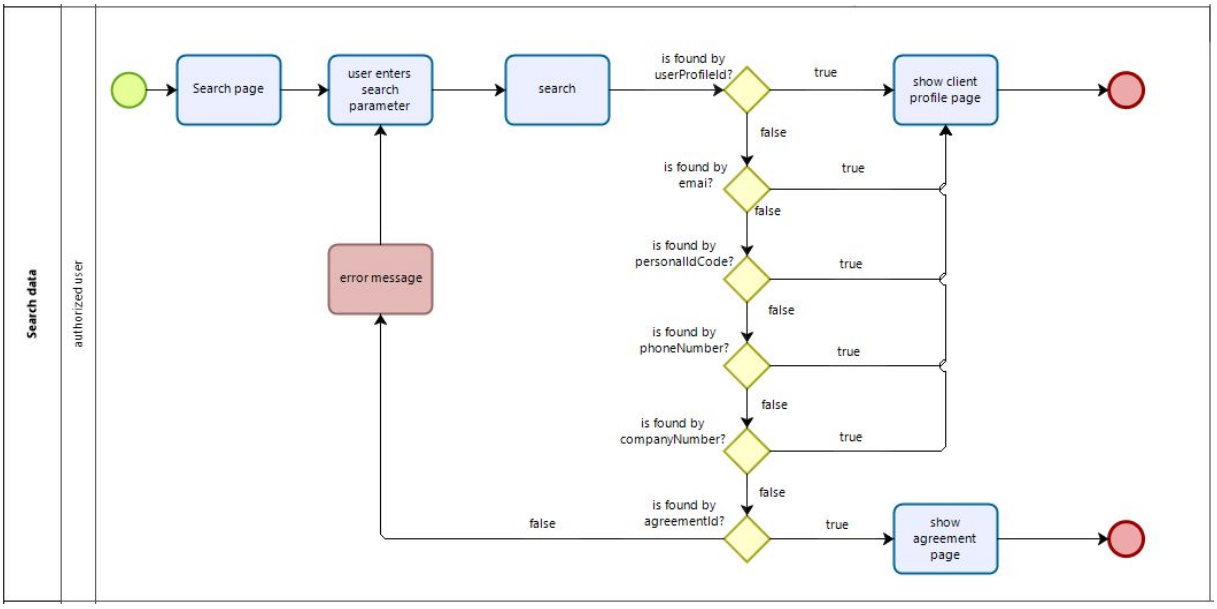
6.3 Kliendi ja lepingu otsimise leht

Otsingu leht on sisse logitud kasutajale peamiseks leheks. Otsingu abil saab leida kliendiandmed ja tema lepingud või konkreetset lepingut. Kliendi otsimise tunnused on

järgmised: isikukood, telefoni number, viitenumber või e-post. Lepingu otsingu tunnus on lepingu number. Kui kasutajat või lepingut ei leita, kuvatakse vastav teave.

Joonis 20. Otsing.

Otsingu loogika on implementeeritud niimoodi, et alguses toimub otsingu väärtuse valideerimine: kas on email, kas on number, kas on tekst, mis sisaldab mingit alam teksti. Peale valideerimist toimub selle otsingu väärtuse kaudu otsing. Allpool on joonistatud otsingu loogika skeem:



Joonis 21. Otsingu loogika skeem.

Allpool on toodud näide, kuidas on implementeeritud otsing, kui otsingu väärtus on number.

```
if (isNumber(searchData)) {
    //try by personal id code
    await this.findProfile( collectionName: 'profiles',  searchValue: "personalIdCode", searchData);
    //try find by mobile phone
    if (!this.props.profile) {
        await this.findProfile( collectionName: 'profiles',  searchValue: "phoneNumber", searchData);
    }
    // try by company number
    if (!this.props.profile) {
        await this.findProfile( collectionName: 'profiles',  searchValue: "companyNumber", searchData);
    }
    this.cleanSearchBar();
    return;
}
```

Joonis 22. Implementeeritud numbri järgi otsingu loogika.

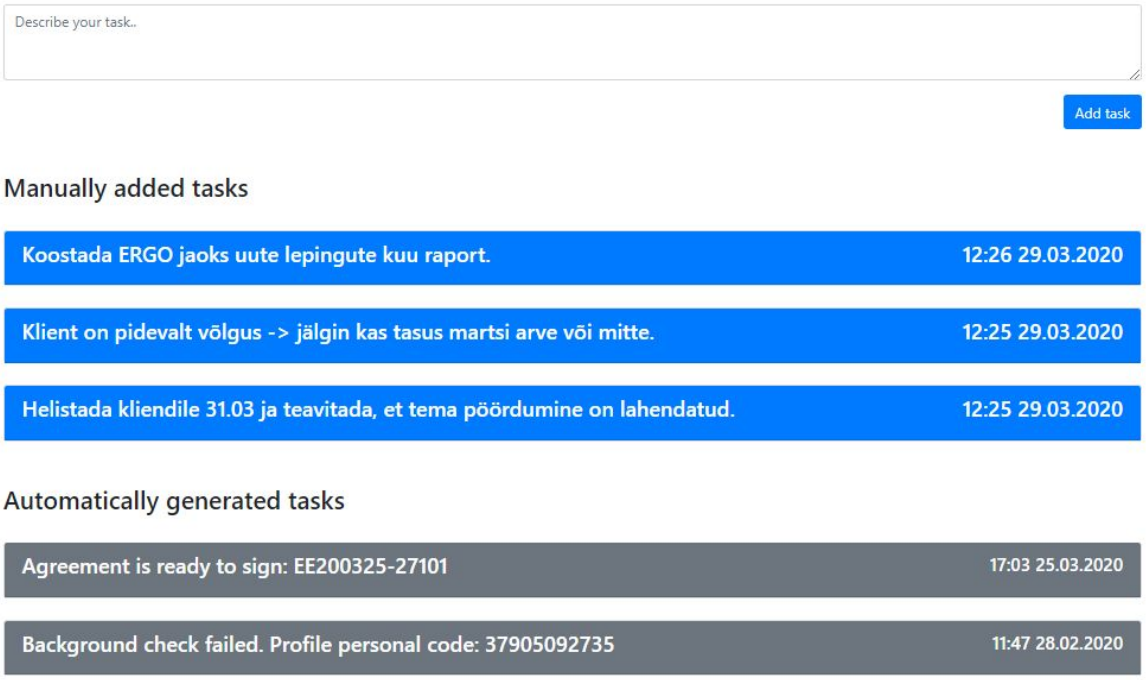
Andmebaasist andmete lugemiseks kasutatakse Cloud Firestore programmi, mille kaudu saab otsida vastavas kollektsioonis(tabelis) vajaliku tunnuse järgi andmed.

```
export async function findCollection(collectionName, searchValue, operator, seacrhData) {
    try {
        const db = database.firestore();
        let data = [];

        if (seacrhData !== null) {
            await db
                .collection(collectionName) CollectionReference
                .where(searchValue, operator, seacrhData) Query
                .get() Promise<QuerySnapshot>
                .then(querySnapshot => {
                    data = querySnapshot.docs.map(doc => {
                        let newOject = doc.data();
                        newOject.id = doc.id;
                        return newOject;
                    });
                });
            return data;
        }
    } catch (err) {
        console.log('Opps, an error occurred', err);
    }
}
```

Joonis 23. Andmebaasi poole päring.

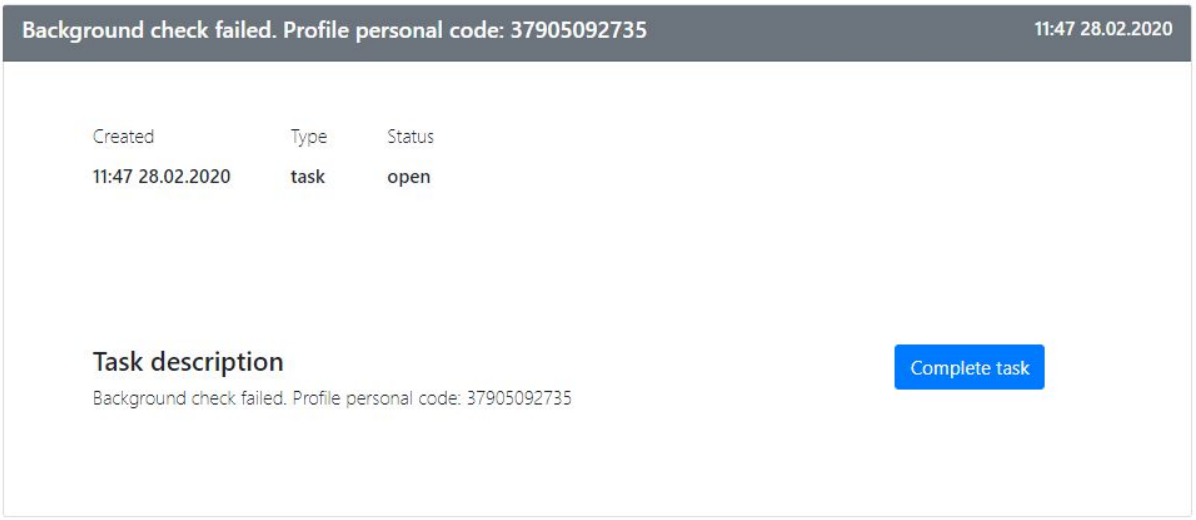
6.4 Ülesannete leht

Ülesannete lehel asuvad kasutaja poolt tehtud ülesanded/meeldetuletused koos kirjelduse ja täitmise kuupäevaga. Peale selle on seal automaatselt genereeritud informatiivsed ülesanded. Loendis kuvatakse loomise aja järgi järjestatud ülesanded. Ülesanne võib-olla kas seotud konkreetse lepinguga või vaba tegum. Vaba ülesannet saab lisada samal lehel kirjutades lahtrisse “kirjeldus”. Lepinguga seotud ülesandeid saab luua ainult avatud lepingus.



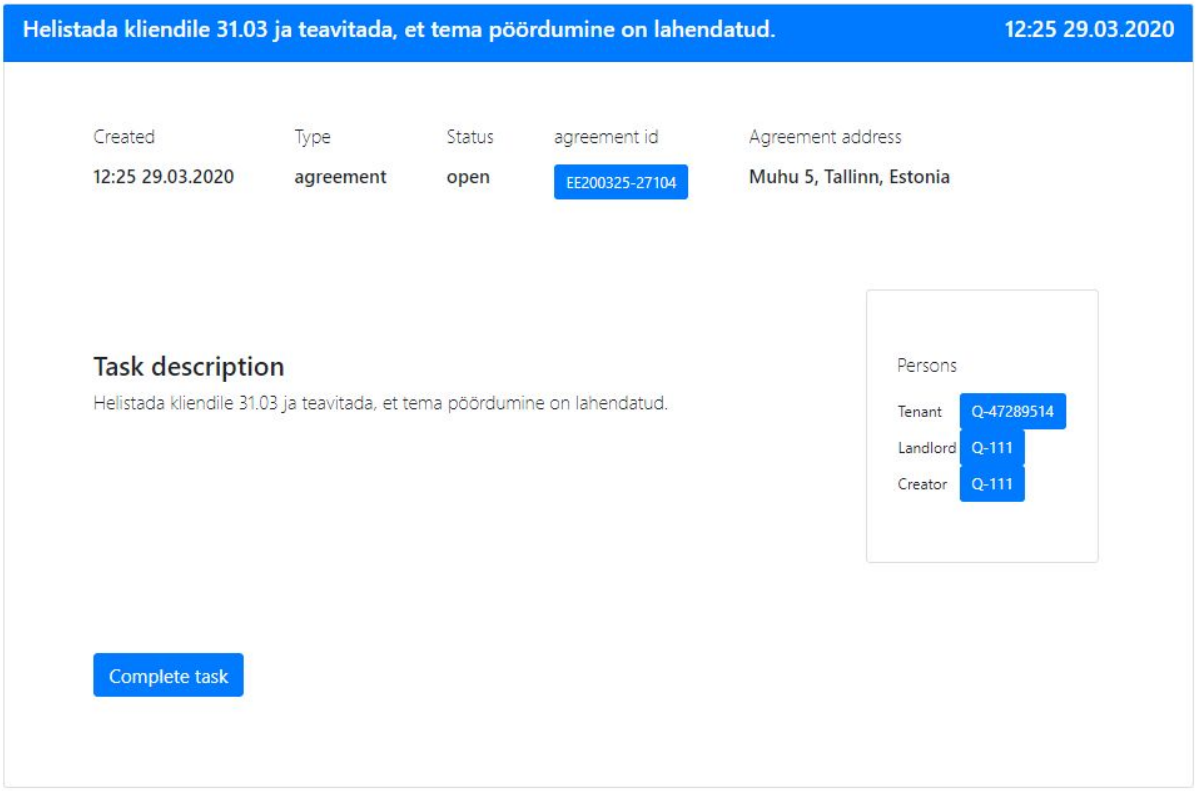
Joonis 24. Käsitsi lisatud ja automaatselt loodud ülesanded.

Automaatselt luuakse tegevused siis, kui on vaja edastada mingi info klienditeenindajale. Automaatne ülesanne tekib näiteks siis, kui registreeritud kasutaja ei läbinud krediidikontrolli või kui leping muutis staatus “allkirjastamiseks valmis” ja nii edasi. Ülesande detailne vaade näitab kirjeldust ja loomise kuupäeva. Tegevust saab sulgeda, kui see on lõpule viidud.

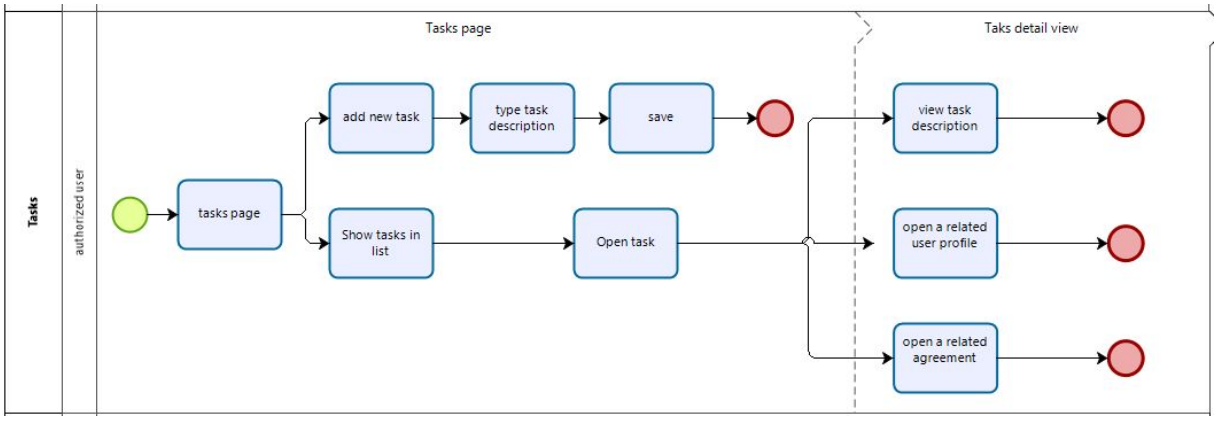


Joonis 25. Automaatselt loodud teade, et krediidikontroll ei läinud läbi.

Iga ülesannet saab avada ja vaadata selle detailset kirjeldust. Detailne vaade näitab loomise kuupäeva, kirjeldust, lepingu numbri ja kolme osapoolt, mis on seotud antud lepinguga. Saab mugavalt liikuda ülesandest lepingu või kliendi andmete vaatesse, vajutades vastavat nuppu ülesande detailses kirjelduses. Ülesande lõpetamise korral on võimalik see sulgeda, vajutades "Lõpeta tegevus" ja antud tegevus kaob loendist.



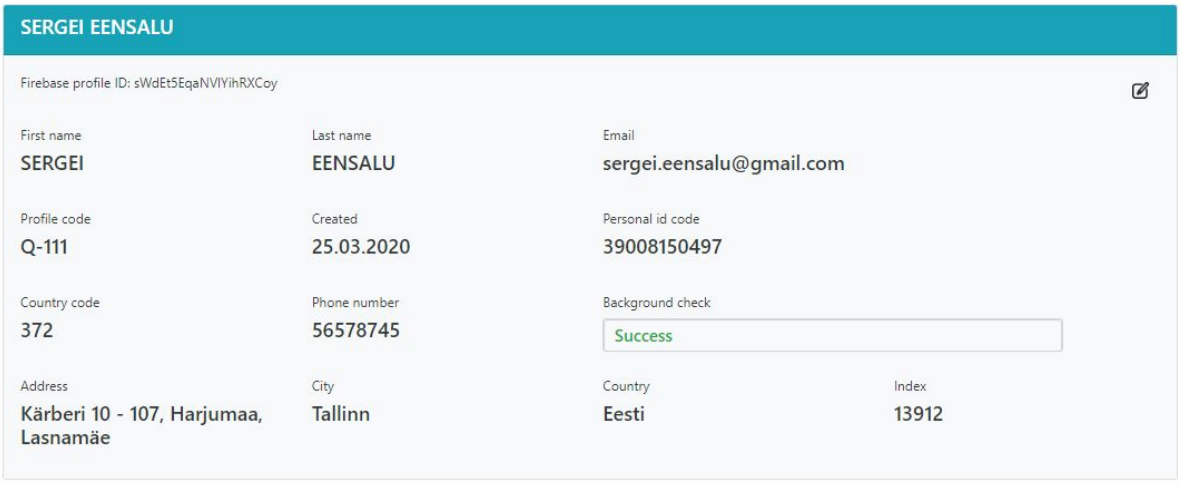
Joonis 26. Avatud ülesanne vaade.



Joonis 27. Ülesanne vaatamine, loomise ja lõpetamise loogika skeem.

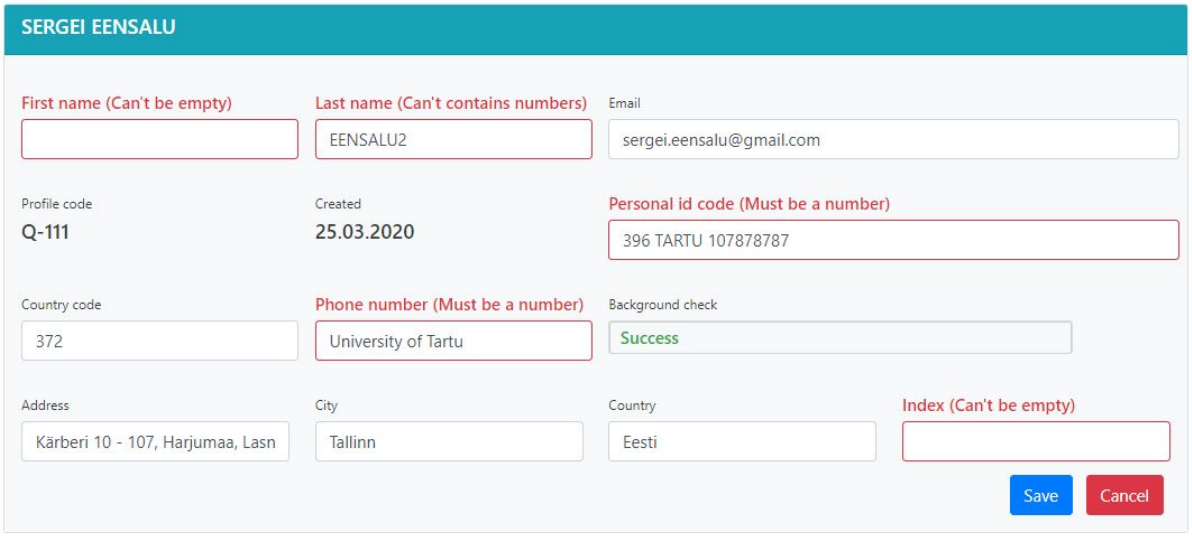
6.5 Kliendi andmete leht

Kliendi andmete lehel asuvad avatud kliendi profiili andmed ja selle kliendiga kõik lepingud. Antud lehel saab vaadata ja muuta kliendi andmed. Samuti võib ka avada seotud kliendiga olevaid lepinguid. Kliendi andmetest on toodud: ees- ja perekonnanimi, isikukood, kontaktnumber ja e-posti aadress, kliendi aadress, krediidiinfo staatus, profiili loomise kuupäeva ja tema profiili number.



Joonis 28. Kliendi profiili andmete vaade.

Andmed saab vajadusel muuta vajutades paremal üleval nurgas vastavat nuppu. Nuppu vajutades, lahtrid muutuvad aktiivseks ja saab täiendada või muuta. Salvestamise käigus toimub kõikide lahtrite andmete valideerimine. Kasutaja saab vastava unikaalse veateade iga lahtri juures, kui validatsioon ei läinud läbi. Muuta ei saa profiili numbrit ja profiili loomise kuupäeva. Krediiti reitingu puhul kasutaja saab avada lisaakna ja vajutades rohelist nuppu, et muutuks krediidiinfo staatus roheliseks.



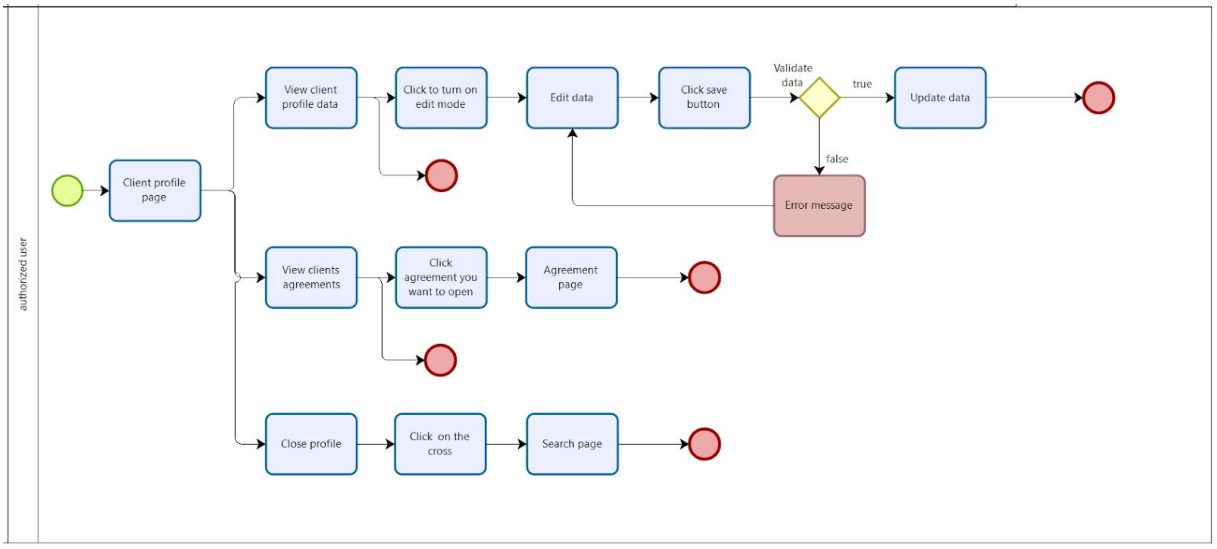
Joonis 29. Kliendi profiili andmete muutmine mitte eduka valideerimise korral.

Kasutaja näeb kõikide antud kliendiga seotud lepingute nimekirja. Nimekiri on kuvatud tabelina, kus on näha lepingu number, aadress, staatus ja avatud kliendi roll antud lepingus. Vajutades vastava lepingu peale avaneb detailne lepingu vaade.

Contracts

Contract id	Address	Status	Role
EE200325-27104	Muhu 5, Tallinn, Estonia	in_review	tenant
EE200325-27101	Testi tänav , Testlinn, EE	ready_to_sign	landlord
EE200325-27102	street, city, EE	draft	creator
EE200325-27101	Testi tänav , Testlinn, EE	ready_to_sign	creator
EE200326-27146	street, city, EE	draft	creator
EE200325-27121	Viies vist , city, EE	archived	creator
EE200325-27103	Kolmas, city, EE	finished	creator
EE200325-27110	street, city, EE	signed	creator

Joonis 30. Kliendiprofiiliga seotud lepingute nimekiri.



Joonis 31. kliendi andmete lehe loogika skeem.

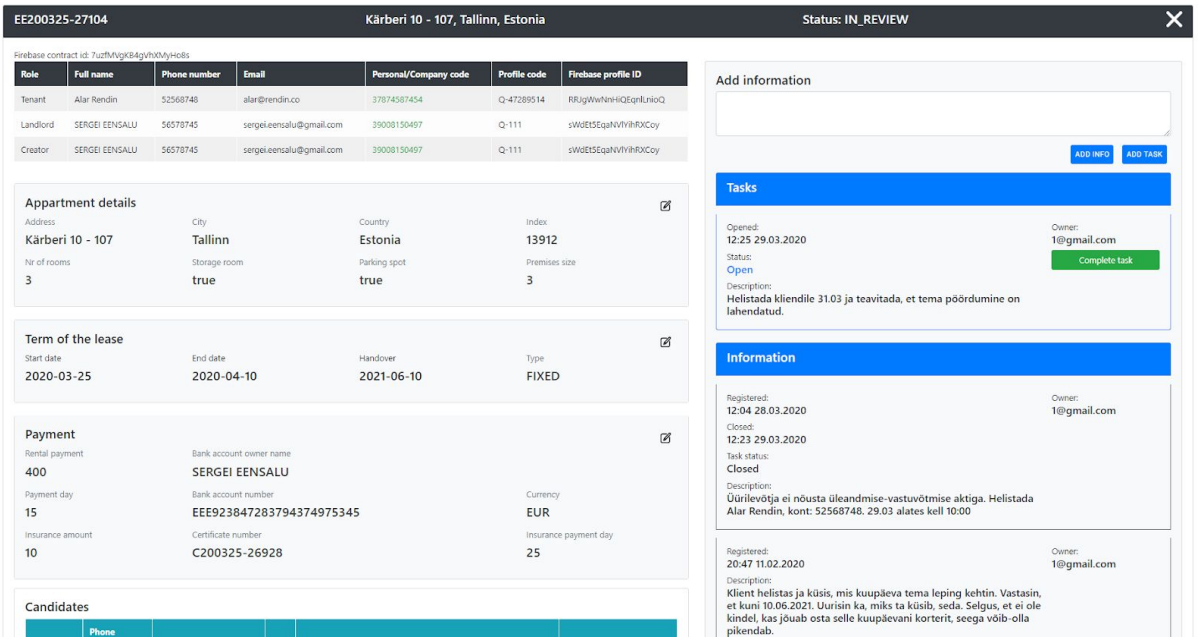
6.6 Lepingu andmete leht

Lepingu andmete lehel asuvad kõik antud lepingu andmed, sündmused ja failid. Mugavuse mõttes, kõik andmed on grupeeritud plokkideks.

Leiduvad järgmised plokid:

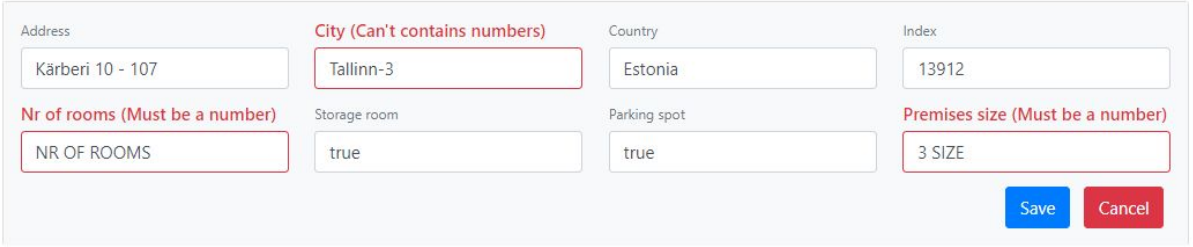
- Lepingu osapooled (üürileandja, üürilevõtja, lepingu looja koos vastava profiili viitega);
- Korteri andmed (aadress, tubade arv, parkimise ja keldrikoha olemasolu info);
- Maksmise andmed (üüri summa, pangakonto andmed, maksmise kuupäev, kindlustuse sertifikaadi number);
- Sündmused, toimingud ja ülesanded;
- Kandidaadid antud korteri üürimiseks;
- Erilised tingimused(võtmete arv, lepingu keel, loomade pidamise luba);
- Dokumendid;
- Vahendid ja teenused (kellele esitatakse erinevate teenuste arved ja kes selliste arvete eest maksab);

- Lepingu staatuse muutmise, PDF-faili lepingu ja ERGO raport genereerimise võimalus;



Joonis 32. Lepingu vaade.

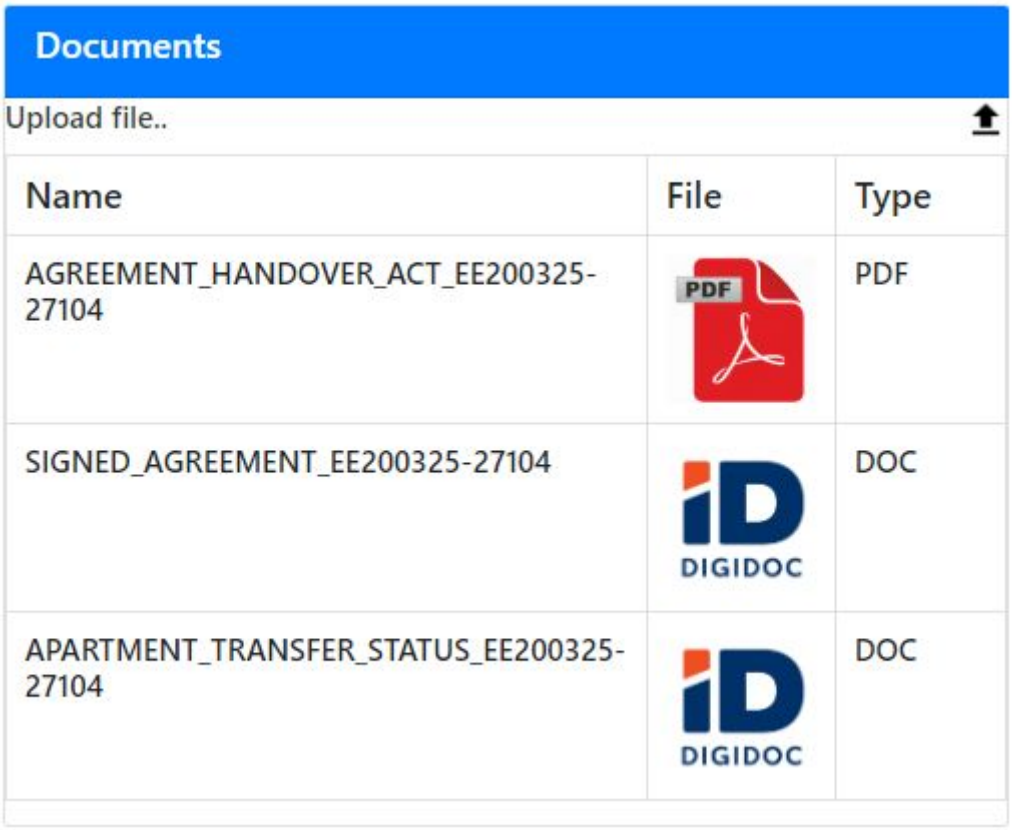
Kasutaja saab soovi korral andmed muuta, vajutades paremal üleval nurgas vastavas plokis nuppu. Toimub andmete validatsioon. Mitte eduka validatsiooni korral avaneb vastav lahter unikaalne veateatega, kus kirjeldab, mida kasutaja poolt oli valesti tehtud. Eduka validatsiooni korral uued andmed salvestatakse andmebaasi.



Joonis 33. Lepingu andmete muutmise mitte eduka valideerimise korral.

Dokumentide plokis asuvad kõik antud lepinguga seotud dokumendid. Kasutaja saab vaadata olemasolevaid dokumente või vajaduse korral lisada uusi faile, vajutades paremal üleval nurgas vastavas plokis nuppu ja valides vajalikud failid kõvakettalt. Dokumendi ülevaatamiseks jaoks, kasutaja võib vajutada vastava dokument peale ja dokument avatakse

eraldi aknas. Failide üleslaadimiste formaadid on ‘.JPG’, ‘.PNG’, ‘,ASICE’, ‘.DDOC’, ‘.BDOC’.



Joonis 34. Lepingu andmete muutmise mitte eduka valideerimise korral.

Kandidaatide kategoorias asuvad kõik antud korterisse kandideerinud kliendiprofiilid. Kandidaadid on kuvatud tabelina, kus on näha inimese täisnimi, telefoninumber, e-post, kodulooma olemasolu, kommentaar ja antud profiili andmebaasi number. Tabeli all asub link, mille kaudu saab antud korterisse kandideerida.

Joonis 35. Kandidaatide nimekiri.

Vahendite ja teenuste kategoorias on määratud vastava teenuse lepingu omanik, arve edastamise kanal ja teenuse eest maksja. Iga teenuse juures saab määrata, kas teenuse lepingu omanik on üürileandja või üürilevõtja, kellele saadetakse arve ja kes selle arve eest maksab.

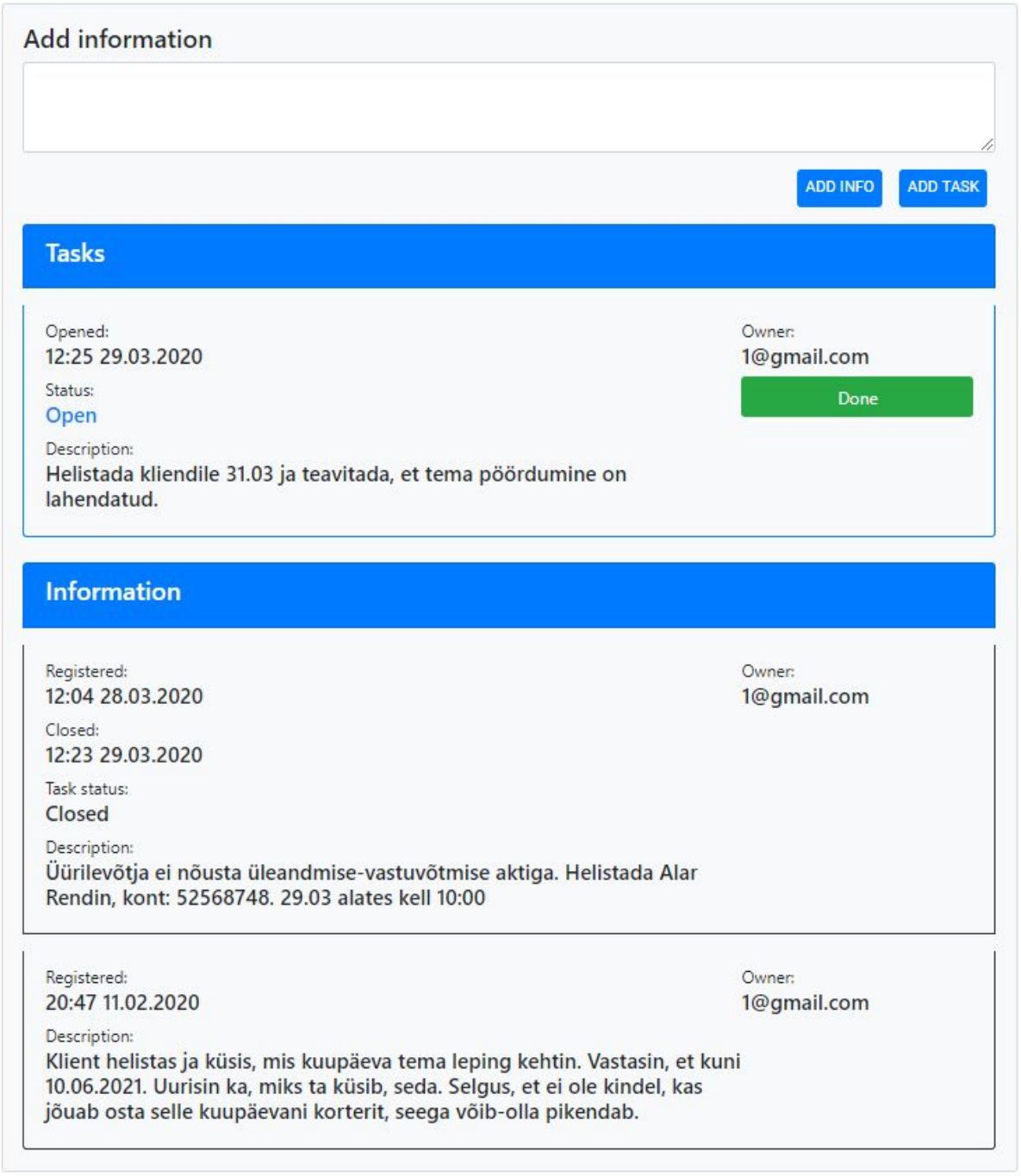
Utilities and Services

Parameter	Contract owner	Who is paying	Payment recipient
Administration fee	landlord	tenant	landlord
Cable tv	tenant	tenant	service provider
Electricity	landlord	tenant	landlord
Gas	tenant	tenant	service provider
General electricity	landlord	tenant	landlord
General water	landlord	tenant	landlord
Heating	landlord	tenant	landlord
House insurance	landlord	tenant	landlord
Insurance	tenant	tenant	service provider
Internet	tenant	tenant	service provider
Land tax	landlord	landlord	service provider
Loan repayment	landlord	landlord	service provider
Maintenance fee	landlord	tenant	landlord
Newspapers magazines	tenant	tenant	service provider
Phone	tenant	tenant	service provider
Repair fund	landlord	landlord	landlord
Security services	tenant	tenant	service provider
Trash	landlord	tenant	landlord
Water	landlord	tenant	landlord

Joonis 36. Vahendite ja teenuste nimekiri.

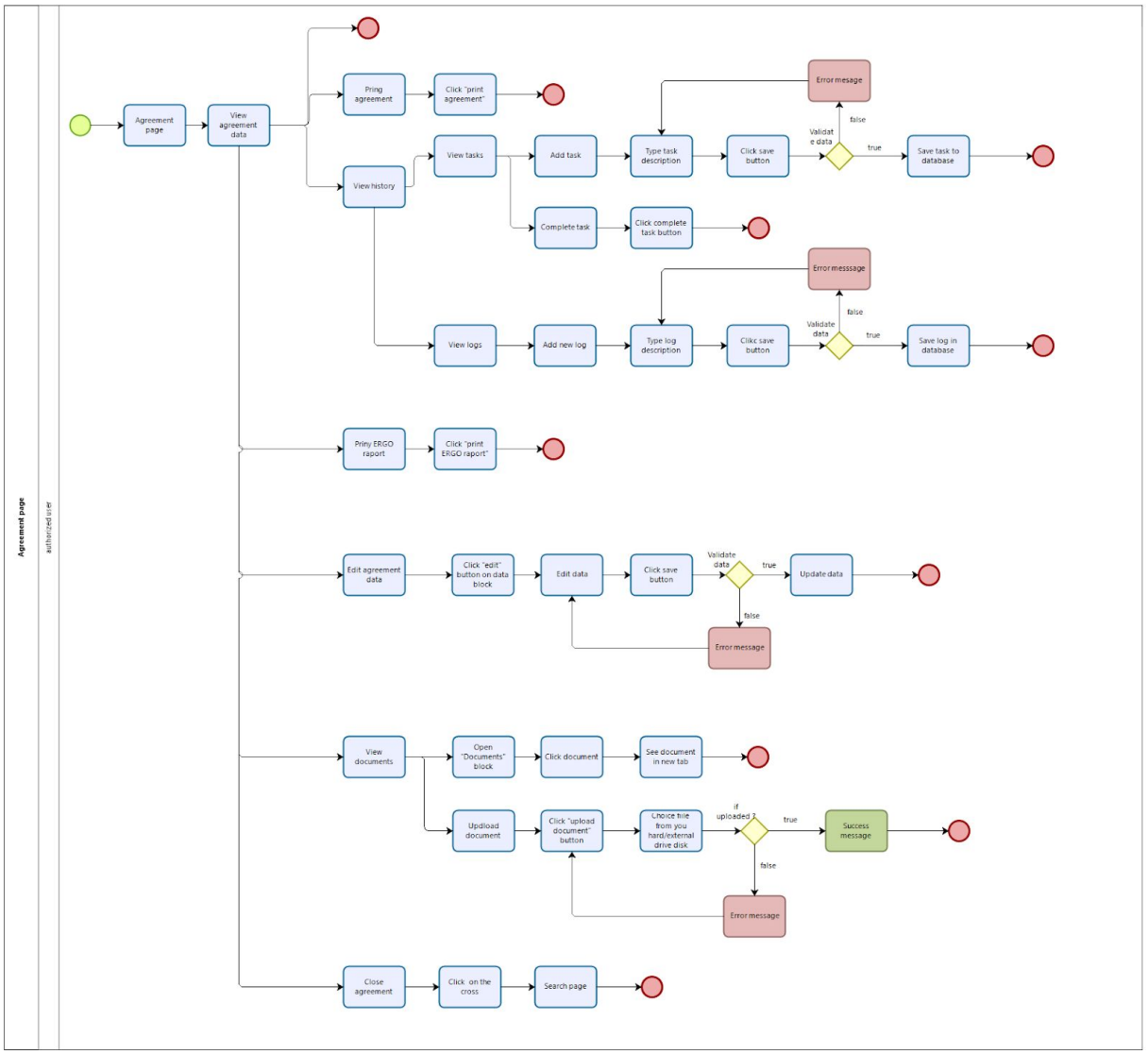
Lepingu kategoorias leidub selle lepingu ajalugu, sündmused ja ülesanded. Lepingu ajaloos on näha kõik muudatused, mis olid lepingus tehtud. Lepingu ajaloosse tekib automaatselt kirje kui lepingus mingi andmed muudetakse või kunagi muudeti. Kasutaja saab luua ka sündmuse, ehk registreerida kliendi pöördumise, kirjutades vajalik informatsioon vastavasse plokki ja vajutades nuppu “lisa sündmus”. Kasutaja saab soovi korral lisada ka ülesande, mis tähendab seda, et antud lepinguga mingi tegevus jäi pooleli ja vajab lahendamist. Ülesannete lisamise protsess käib samamoodi nagu sündmuste lisamisel, kasutaja kirjutab ülesande kirjelduse ja vajutab nuppu “lisa töö”. Tekib ülesanne, mis on esimesena nähtav lepingu küljes, kõikide teiste ülesannete nimekirjas. Iga töö saab muuta lõpetatuks vajutades nuppu “Tehtud”.

Kasutaja mugavuse tagamiseks nii ülesande kui ka sündmuse puhul on kuvatud selle tegevuse looja e-post, loomise ning lõpetamise kuupäev, kirjeldus ja staatus.



Joonis 37. Lepingu sündmuste ja ülesannete vaade.

Lepingu ülevaatamine ja muutmine on üks kõige olulisem osa antud tarkvara arenduses, seega kõige rohkem funktsionaalsust asub avatud lepingu vaates.



Joonis 38. Lepingu lehe loogika skeem.

7. Kokkuvõte

Antud töö eesmärk oli luua kasutaja haldamise tarkvara, mis aitab ettevõtte töötajatele käsitleda kliendi ja lepingu andmeid.

Ettevõttes puudus turvaline ja mugav viis kiiresti kliendi ning lepingu andmete otsimiseks, nende vaatamiseks ja muutmiseks. Andmete otsimine, vaatamine ja muutmine toimus ainult otse andmebaasist töötaja poolt, kellel puudus andmebaasidega töötamise kogemus. See protsess vajas kiiret, turvalist ja mugavat lahendust.

Antud bakalaureusetöö raames oli tuvastatud ettevõtte jaoks haldamise tarkvara vajadus, tehtud selle tarkvara analüüs, püstitatud eesmärgid ja nõuded ning tehtud kokkulepped kasutatavate tehnoloogiate osas. On arendatud ReactJS projekt, testitud ja dokumenteeritud.

Kõik funktsionaalsed ja mittefunktsionaalsed nõuded on täidetud. Kliendi soovidega on arvestatud. Antud bakalaureusetöö raames loodud kasutaja haldamistarkvara on registreeritud domeenile (https://rendin-production.firebaseapp.com/). Valmis tarkvara on kliendi poolt aktsepteeritud ja kasutusel.

Kliendiga koostöö jätkub. Antud töö autor sõlmib töösuhted Rendin OÜ idufirmaga ja asub tootehalduse ja tarkvaraarenduse ametikohale, kus hakkab vastutama kasutaja haldamise, tarkvara arenduse ja haldamise eest ning sisestab süsteemi firma poolt antud tööülesandeid.

8. Kasutatud allikad

[1] *Rendin OÜ firma kirjeldus ja korduma kippuvad küsimused.* Kättesaadav: https://et.rendin.co/faq (10.01.2020)

[2] Zeltserman K. B. *Kliendibaasi optimeerimine. Müügijuhtimine nr 10 (2005).* Kättesaadav: http://www.axima-consult.ru/stati-optimkb.html (15.01.2020)

[3] Maria Smirnova-Matros. *Mis on kliendibaas?* (28.05.2018) Kättesaadav: https://www.unisender.com/ru/support/about/glossary/chto-takoe-baza-klientov/ (15.01.2020)

[4] Class365.ru *Tõhusad viisid kliendibaasiga töötamiseks.* Kättesaadav: https://class365.ru/crm/rabota-s-klientskoi-bazoi (15.01.2020)

[5] Customer-support.financesonline.com *10 Best Customer Support Software Systems For Your Company. (10 parimat ettevõtte klienditoe tarkvarasüsteemi.)* Kättesaadav: https://customer-support.financesonline.com/top-10-customer-support-software-for-your-company/ (15.01.2020)

[6] Allclients.com *Customer Database Management / Client Online Database. (Kliendiandmebaasi haldus / kliendi veebiandmebaas).* Kättesaadav: https://allclients.com/client-database/ (15.01.2020)

[7] City24.ee ettevõtte kirjelduse ja kontaktide leht. Kättesaadav: https://www.city24.ee/et/contacts/ (15.01.2020)

[8] Kinnisvara24 ettevõtte kirjelduse ja kontaktide leht. Kättesaadav: https://domuskinnisvara.ee/2018/04/18/alustas-uus-kinnisvaraportaal-kinnisvara24/ (15.01.2020)

[9] KV.EE ettevõtte kirjelduse ja kontaktide leht. Kättesaadav: https://www.kinnisvaratallinnas.ee/tutvustus (10.02.2020)

[10] JavaScript'i raamistiku ja versioonide kirjeldus. Kättesaadav: https://developer.mozilla.org/en-US/docs/Web/JavaScript/About_JavaScript (15.03.2020)

[11] Lauren Schaefer. NoSQL andmebaasi definitsioon ja kirjeldus. Kättesaadav: https://www.mongodb.com/nosql-explained (23.03.2020)

[12] Wikipedia *CSS*. (2018) Kättesaadav: https://et.wikipedia.org/wiki/CSS (01.04.2020)

[13] Newline *What is React? (Mis on ReactJS?)* Kättesaadav: https://www.newline.co/fullstack-react/30-days-of-react/day-1/ (08.04.2020)

[14] Wikipedia *Bootstrap (front-end framework) (kujunduse raamistik) Kättesaadav:* https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework) (08.04.2020)

[15] React *Intro to React (Sissejuhatus ReactJS).* Kättesaadav: https://reactjs.org/tutorial/tutorial.html (10.04.2020)

[16] Chris Parker. File *Structure for React Applications. Created with create-react-app.* (ReactJS projekti failide struktuur. Luua projekti create-react-app käsu kaudu.) (15.08.2019). Kättesaadav: https://www.pluralsight.com/guides/file-structure-react-applications-created-create-react-app (10.04.2020)

[17] Wikipedia *Redux*. Kättesaadav: https://et.wikipedia.org/wiki/Redux_(JavaScripti_raamistik) (20.04.2020)

[18] *Redux* toimingute ametlik dokumentatsioon ja kirjeldus. Kättesaadav: https://redux.js.org/basics/actions/ (22.04.2020)

[19] *Redux* reduktorite projektiga sidumise ametlik dokumentatsioon ja juhend. Kättesaadav: https://redux.js.org/basics/reducers (25.04.2020)

[20] *Redux* ja *ReactJS* komponentide sidumise ametlik dokumentatsioon ja juhend. Kättesaadav: https://react-redux.js.org/api/connect (29.04.2020)

[21] *Firebase Security Rules* (Firebase turvalise õigused) (08.04.2020) Kättesaadav: https://firebase.google.com/docs/rules (29.04.2020)

[22] *Firebase* ja *JavaScript'i* andmebaasi ühenduse loomise ametlik dokumentatsioon ja judend. Kättesaadav: https://www.npmjs.com/package/firebase (30.04.2020)

9. Lisad

I. Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks

Mina, **Sergei Eensalu**,

(autori nimi)

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) minu loodud teose

   Kasutaja haldamise tarkvara idufirmas *Rendin OÜ*,

   (lõputöö pealkiri)

   mille juhendaja on **Vambola Leping**,
   (juhendaja nimi)

   reprodutseerimiseks eesmärgiga seda säilitada, sealhulgas lisada digitaalarhiivi DSpace kuni autoriõiguse kehtivuse lõppemiseni.

2. Annan Tartu Ülikoolile loa teha punktis 1 nimetatud teos üldsusele kättesaadavaks Tartu Ülikooli veebikeskkonna, sealhulgas digitaalarhiivi DSpace kaudu Creative Commonsi litsentsiga CC BY NC ND 3.0, mis lubab autorile viidates teost reprodutseerida, levitada ja üldsusele suunata ning keelab luua tuletatud teost ja kasutada teost ärieesmärgil, kuni autoriõiguse kehtivuse lõppemiseni.

3. Olen teadlik, et punktides 1 ja 2 nimetatud õigused jäävad alles ka autorile.

4. Kinnitan, et lihtlitsentsi andmisega ei riku ma teiste isikute intellektuaalomandi ega isikuandmete kaitse õigusaktidest tulenevaid õigusi.

Sergei Eensalu
07.05.2020