TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Kristjan Pille 186061IADB

FÜÜSILISE TÖÖ MONITOORINGUSÜSTEEMI LOOMINE

Bakalaureusetöö

Juhendaja: Kaido Kikkas
Tehnikateaduste doktor

Tallinn 2021

Autorideklaratsioon

Kinnitan, et olen koostanud käesoleva 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: Kristjan Pille

20.02.2021

Annotatsioon

Käesoleva bakalaureusetöö eesmärgiks on luua kergesti käsitletav veebirakendus raskuste käsitsi teisaldamise terviseriskide hindamiseks töökeskkonnas. Töötajatel ja tööandjatel puudub kergesti ja mugavalt kasutatav elektroonne töövahend eelpool nimetatud terviseriskide kvantitatiivseks hindamiseks.

Käesolevas töös luuakse mugavalt käsitletav rakendus raskuste käsitsi teisaldamise terviseriskide hindamiseks ning varajase sekkumise võimaldamiseks. Lisaks vormi täitmisele, mis arvutab selle kasutajale automaatselt lõpptulemuse, on tulemusi võimalik salvestada ning neid hiljem analüüsida, et tekiks terviklik pilt töökeskkonna üldisest seisukorrast. Eelkõige on rakendus mõeldud töökeskkonnaspetsialistidele ning ergonoomidele, kuid ka tööandjatele ja töötajatele, mis võimaldab kontrollida, kas töökeskkond on vaba võimalikest terviseriskidest.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 27 leheküljel, 5 peatükki, 10 joonist.

Abstract

Development of a Monitoring System for Manual Labour

The goal of this Bachelor's Thesis is to develop an easily managable web application for the assesment of physical workloads. As currently labourers and employers lack access to the digitalized way of assessing working conditions.

The aim of the study is to develop physical workload assessment application. In addition to digitalized form filling that automatically calculates the end result for the user, it is possible to save the results and analyze them to get a more in-depth overview of the given work environment. First and foremost the application is meant for ergonomist and work environment specialists, but it can also be used by employers and employees to get approximate safety assessment of their work environment.

The thesis is written in Estonian and contains 27 pages of text, 5 chapters, 10 figures.

Lühendite ja mõistete sõnastik

API	Application Programming Interface – rakenduse programmerimise liides
Baua	Federal Institute of Occupational Safety and Health - Saksamaa Riiklik Töötervishoiu Instituut
CRUD	Loomine (Create), lugemine (Read), uuendamine (Update), kustutamine (Delete). Põhifunktsioonid andmebaasirakenduses.
ERD	Entity Relationship Diagram – Andmete struktuuri kirjeldav skeem
i18n	Internatsionaliseerimine
HIKO	Hinda käsitsi teisaldamise ohutust
HTTP	HyperText Transfer Protocol – hüperteksti edastusprotokoll
JSON	JavaScript Object Notation – Javascript'il põhinev andmevahetusvorming
KIM	Võtmeindikaatori meetod füüsilise töökoormuse hindamiseks ja kavandamiseks
LUBA	Kehaasendi põhine ülakeha koormuse hindamise metoodika
MVC	Model-View-Controller – Mudel-Vaade-Kontroller
REST	Representational State Transfer – tarkvara arhitektuuri disain
RKT	Raskuste käsitsi teisaldamine
RULA	Kogu keha ja käte koormuse kiirhindamise metoodika
SQL	Structured Query Language – struktureeritud päringukeel

Sisukord

1 Sissejuhatus 9
Probleemi ülevaade 11
1.1 Olemasolevad lahendused 12
1.1.1 Töö füsioloogilise koormuse hindamise küsimustikud 13
1.1.2 Digitaliseeritud lahendused 13
2 Projekti skoop 15
3 Veebirakenduse analüüs 16
3.1 Andmebaaasi valik 16
3.2 Tagarakenduse tehnoloogia valik 18
3.2.1 Tagarakenduse programmeerimiskeele valik 18
3.2.2 Raamistiku valik 19
3.2.3 Kasutajaliidese ühendus 19
3.3 Eesrakenduse raamistiku valik 20
3.3.1 Vue 20
3.3.2 React 20
3.3.3 Angular 20
3.4 Arenduskeskkonna valik 21
3.4.1 Versioonihaldustarkvarad 21
3.4.2 Integreeritud arenduskeskkonna valik 22
4 Arendus 24
4.1 Tagarakenduse arendus 24
4.1.1 ASP.NET Core 24
4.1.2 Tagarakenduse ülesehitus 25
4.1.3 Andmebaasi kavandamine 26
4.2 Eesrakenduse arendus 27
4.2.1 ReactJS teek 27
4.2.2 Eesrakenduse ülesehitus 27
4.2.3 Rakenduse disain 28
4.3 Rakenduse turvalisus 28
4.4 Valminud veebirakenduse analüüs 28
5 Rakenduse realisatsioon 30

5.1 Kasutajatuvastus 30
5.2 Vormi täitmine 31
5.3 Vormi täitmise tulemused 31
5.4 Salvestatud tulemused 32
5.5 Ettevõtted 32
5.6 Vormi valideerimine 33
5.7 Kasutajate haldus 34
6 Tulevikuarendused 35
Kokkuvõte 36
Kasutatud kirjandus 37
Lisa 1 – Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks 41

Jooniste loetelu

Joonis 1. Spring MVC Mudel. 18
Joonis 2. .NET 5 raamistiku mudel. 24
Joonis 3. Vormi kontrolleri lõpp-punktid 25
Joonis 4. Andmebaasi olemi-suhte diagramm. 26
Joonis 5. Rakenduse sisselogimisleht. 30
Joonis 6. Vormi algus. 31
Joonis 7. Tulemuse vaade. 32
Joonis 8. Töötajate vormid. 32
Joonis 9. Ettevõtte põhine vaade. 33
Joonis 10. Vormi valideerimise näide. 34

1 Sissejuhatus

Väga paljudes töövaldkondades nagu ehitus, transport, põllumajandus, metsandus, tervishoid esineb rasket füüsilist tööd. Luu-lihaskonna haigused on juhtivaks tööst põhjustatud haiguste diagnoosi grupiks [1].

Tööga seotud luu-lihaskonna haiguste ennetamiseks on vajalik hinnata töökoormust töökohtadel. See nõue on kehtestatud kõigis Euroopa Liidu liikmesriikides töötervishoiualase seadusandlusega (OSHA Wiki, RKT määrus Eesti) [1] [2].

Arvutuslikult on hinnatud, et töötervishoiule ja tööohutusele panustatud kulutused vähendavad tervisekahjudest põhjustatud kulusid ja suurendavad tootlikust. Vastavalt Chapmani meta-analüüsile (2005) võivad hästi juhitud töötervise edendusprogrammid vähendada 27% haigestumiste tõttu töölt eemal viibimist, 26% tervishoiu kulusid ja 32 % kindlustusele tehtavaid kulutusi [2].

KIM (*Key indicator method*) meetod on täiendatud ja välja arendatud universaalseks raskuste käsitsi teisaldamise hindamise meetodiks Saksamaa Riikliku Töötervishoiu Instituudi Baua poolt 2001 aastal ning uuendatud 2019 aastal. Uudne meetod sobib töökoormuse hindamiseks käsitsi tõstmise, hoidmise ja teisaldamisega seotud tööde korral. Meetod sobib raskuste korral, mille kaal algab kolmest kilogrammist ning on universaalne ja kergesti kasutatav nii töökeskkonnaspetsialistidele, ergonoomidele kui ka tööandjatele, kes ise riskianalüüsi läbi viivad [3].

Käesoleva töö eesmärgiks on luua kergesti käsitletav digitaalse töövahendi rakendus raskuste käsitsi teisaldamise terviseriskide hindamiseks ning varajase sekkumise võimaldamiseks, et vältida kroonilisi tööst põhjustatud haigusi nagu krooniline seljavalu, radikulopaatiad jm.

Bakalaureusetöö jaguneb viieks peatükiks. Esimeses peatükis tutvustatakse töö kirjutamise hetkel aktuaalset töötervishoiu süsteemi digilahenduste probleemi ning tuuakse välja olemasolevad alternatiivsed lahendused ja nende analüüs. Teises peatükis

antakse ülevaade loodavast veebirakendusest ning käesoleva töö skoobist. Kolmandas peatükis analüüsitakse andmebaasi ning ees-ja tagarakenduse tehnoloogiate valikut. Selgitatakse arenduskeskkondade valikute printsiipe ning põhjendatakse lõplikku valikut. Neljandas peatükis antakse ülevaade ees-ja tagarakenduse arenduse protsessidest. Kavandatakse andmebaasi struktuur ning tuuakse välja rakenduses kasutatav ERD mudel. Samuti tuuakse välja rakenduses kasutusele võetud turvameetmed ning seletatakse lahti nende tähtsus. Neljanda peatüki viimaseks osaks on rakenduse analüüs, kus kirjeldatakse lähemalt valminud rakendust. Viiendas peatükis antakse põhjalik ülevaade valminud rakenduse struktuurist, tuuakse välja rakenduse osad ning nende funktsionaalsus.

Probleemi ülevaade

Füüsiline töökoormus põhjustab tervisekahjustusi ka tänapäeval olles nii ajutise kui püsiva töövõime põhjustajaks luu-lihaskonna haiguste tõttu, mis on koormuseks riigi majandusele. Käsitsi raskuste teisaldamist esineb erinevates tööstusvaldkondades, toitlustuses, kaubanduses jne. [3]

Senimaani kasutatakse Eestis ergonoomide poolt laialdaselt RULA ning LUBA (vt täpsemalt 1.1.1) metoodika pabervorme füüsilise töökoormuse hindamiseks. Lisaks nimetatule on ametlik raskuste käsitsi teisaldamise töötervishoiu ja tööohutuse nõuete valideerimise meetod aastast 2001 ainult pabervormil [2]. Eelnevalt nimetatud meetod koosneb kolmest osast ning annab ainult väga üldise ülevaate töökeskkonnast, mille alusel tulemus arvutatakse. Paljud raskuste teisaldamisega seotud liigutused jäävad selle skoobist väljapoole, mille tulemusena jääb tähelepanuta suur hulk potentsiaalseid terviseriske, mis pikas perspektiivis võivad osutuda vägagi kulukaks töötajale või halvemal juhul kogu töötajaskonnale. Lisaks töötajatele tekitab see ka probleeme tööandjatele nii seadusega kehtestatud korra rikkumise pärast, kui ka töötajatele tervise-kahjustuste eest kompensatsiooni tasumisega.

Töötajatel ja tööandjatel puudub kergesti ja mugavalt kasutatav elektroonne töövahend nimetatud valdkonnas kvantitatiivselt terviseriskide hindamiseks. Praegune metoodika Eestis raskuste käsitsi teisaldamise töötervishoiu ja tööohutuse nõuete valideerimiseks on aastast 2001, mis on pabervormil ning ei ole kõige kaasaegsem ja efektiivsem viis tööohutuse tagamiseks [2].

Eesti ettevõtete töökeskkondades on raskuste käsitsi teisaldamisega seotud tööd laialt levinud. Tööohutuse ja töötajate tervise hoidmise osas on ettevõtete tasemed väga erinevad. Probleemi lahendamisel on abiks käesoleva töö tulemusena välja arendatud veebirakendus, mis on kergesti kätte saadav ja lihtsalt kasutatav.

Lõppkasutaja vaatest on võimalik kohene vormi täitmine ilma oma andmeid sisestamata ning teine vaade, kus kasutaja on sisse logitud ja tal on ligipääs kõigile tema täidetud vormidele ning nende tulemustele. Lisaks kasutajatele on rakenduses eraldi sektsioon mõeldud administratsioonidele, kellel on võimalus registreerida uusi kasutajaid ning neid hallata. Rakenduses vormi täitmisel nõutavate andmete sisestamine võtab aega keskmiselt

kaheksa minutit. Tulemuse arvutab rakendus automaatselt ja esitab tulemuse numbrilise skoori koos kokkuvõtliku nõustava hinnanguga tehtava töö terviseriskide kohta. Andmeid on võimalik salvestada. Töökoha ja töökorralduse muutmise järel on lihtne viia läbi kordushindamine, et veenduda terviseriskide vähenemises. Ühes ettevõttes võib olla töötajatel väga erinevaid teisaldamistöid, näiteks suurtes toidukauplustes on tavapäraselt palju laotööd ja kaupade saali ladustamist. Sellistel juhtudel on lihtne hinnata raskemaid töö etappe, mis on seotud esmahinnanguliselt tervisele ohtliku mõjuga, kuid oluliselt lühema ajaga varasemast saab ülevaate ka kõigist muudest teisaldamisega seotud töödest individuaalselt iga töötaja kohta.

Taoline uudne ja kergesti käsitletav veebirakendus on efektiivsem viis raskuste teisaldamisega seotud tööde ohutuse hindamiseks, kuna jääb ära mitmete pabervormide täitmine ja käsitsi kokku arvutamine. Sageli jäetakse paberkandjal kohmakas hindamisprotsess üldse läbi viimata [4].

Uudne moment on veebirakenduse juures asjaolu, et varasemalt ei ole töötajad saanud ise oma töötingimusi hinnata. Kui arvestada senini kasutusel olevat raskuste käsitsi teisaldamise hindamisvormi, siis tulemus võib kujuneda hindaja põhiselt subjektiivseks ning töötaja või töötajad võivad jääda tulemuste osas eriarvamusele. Uue veebirakendusega arvutatud tulemus võtab aluseks oluliselt rohkem ja täpsemaid andmeid ning on objektiivsem ja võimaldab töötajatel aktiivselt osaleda oma koormuse reguleerimisel.

Pikemas perspektiivis on hindamistulemuste adekvaatse rakendamise korral hoitud töötajate tervis, esineb vähem ülekoormusest tingitud luu-lihaskonna haigestumisi ning suureneb töötajate tööga rahulolu.

1.1 Olemasolevad lahendused

Eestis ja mujalgi Euroopas on laialdaselt kasutusel tööohutegurite ohutaseme hindamiseks Euroopa töötervishoiu ja tööohutuse agentuuri soovituslik maatriks. Hinnatakse riski taset tulenevalt ohuteguri esinemise tõenäosusest ja tervisekahjustuse tagajärgede võimalikust tõsidusest. Esinemise tõenäosus ja tagajärgede raskus on jaotatud kolmeks erinevaks tasemeks: 1) ei ilmne töötamise aja jooksul kordagi, 2) võib esineda

ainult paaril korral, 3) esineb kogu töötamise ajal korduvalt. Tagajärgede tõsidust peetakse väheohtlikuks, kui see ei põhjusta õnnetusi või haigusi. Ohtlikkuse järgmised kaks taset määratakse vastavalt: 1) põhjustab kerge vigastuse või haigestumise 2) põhjustab raske tööõnnetuse või pikaajalise haiguse. Riski tasemed 1 kuni 5: jaotuvad järgnevalt: vähene risk, vastuvõetav risk, keskmine risk, suur risk, talumatu risk. Taoline maatriksil põhinev hindamine toimub subjektiivse vaatluse ja hindaja teadmiste ning kogemuste põhjal. Hindamine ei ole objektiviseeritud numbriliste näitajatega ning sageli ei anna objektiivset tulemust ja on väga palju hindaja oskustest sõltuv. Seetõttu on ka Eestis ettevõtete riskianaüüside madal kvaliteet saanud korduvalt kriitikat Tööinspektsiooni poolt [4].

Täpsemaks füüsilise töökoormuse hindamiseks on loodud mitmeid erinevaid meetodeid, samuti erinevate keha piirkondade (käed, jalad, selg) koormuste hindamiseks [1].

1.1.1 Töö füsioloogilise koormuse hindamise küsimustikud

- LUBA *assessment for postural loading on the upper body* on hindamismeetod, mis on loodud käte ja ülakeha füüsilise töö koormuse hindamiseks [5].

- *Niosh lifting equation* on teisaldamistöö raskuskategooria hindamismeetod, kuid on kohaldatav teatud tingimustel. Ei ole kasutatav kui teisaldamisega kaasneb kandmine pikemat vahemaad, ei sobi inimeste ja loomade siirdamise töö raskuse hindamiseks, kuna meetod eeldab statsionaarseid objekte [5].

1.1.2 Digitaliseeritud lahendused

- Ergo Plus – Vaadeldav ettevõte pakub firmadele töökeskkonna inspektsiooni lahendusi ehk tullakse kohale ning teostatakse vastavalt kliendi soovile määratud inspektsioon [6]. Seda viiakse läbi ainult teenuse osutaja poolt ja tavatöölistel puudub võimalus individuaalseks kasutamiseks. Lisaks Ergo Plussis esindatud meetodite raskuste teisaldamise universaalsuse puudumisele teostab nimetatud ettevõte firmadele lahendusi vastavalt tellimustele ning ei ole kasutatav individuaalselt, vaid ainult Ergo Plus ekspertide poolt. Ergonoomidel, töökeskkonnaspetsialistidel ning tavatöötajatel puudub võimalus seda laialdasemalt kasutada töö käigus. Käesoleva lõputöö raames valmival rakendusel

on suur osakaal kasutajasõbralikkusel, sest lisaks töökeskkonnaspetsialistidele saaksid seda kasutada ka tavatöötajad. Ühtlasi on testi tegemine tasuta ja kõik, kellel soov oma töökeskkonda hinnata, saavad seda teha.

- Osmond Ergonomics – Vaadeldava ettevõtte poolt on koostatud RULA metoodikast digitaliseeritud lahendus. See metoodika mõõdab raskuste koormust ainult ülakehale ning ei võta arvesse teisi tähtsaid aspekte, mis võivad raskuste teisaldamisega kaasneda. Vormi täitmine lehel on kasutajakogemuse poolest kasin. [7]

2 Projekti skoop

Loodav veebirakendus on lõputöö raames kasutatav eestikeelsena. Aluseks võetud vormil on tõlkeid mitmes erinevas keeles ning vajadusel on võimalus lisada nende keelte tugi, et valminud rakendust oleks võimalik kasutada ka väljaspool Eestit, kuid esialgu jääb see lõputöö skoobist väljapoole. Rakendus on kasutatav sülearvutites, personaalarvutites ja nutiseadmetes.

Valmis kujul on digitaliseeritud vorm veebirakenduses kasutusvalmis ka tavatöötajatele. Kohese tulemuse alusel, mille rakendus välja arvutab, on võimalik vastavalt tulemuse skoorile edasi tegutseda. Rakenduse teeb universaalseks võimalus, et seda saab kasutada täiesti tasuta, ning eriala spetsialistidele või tööandjatele, kes soovivad tulemusi salvestada ning edasi vajadusel analüüsida, on eraldi ala rakenduses, kus saab ülevaate täidetud vormidest ning nende tulemustest. Lisaks on võimalus kategoriseerida tulemusi ettevõtete kaupa ning iga ettevõtte kohta on võimalus omakorda kategoriseerida andmed ametite gruppide kaupa (laotöölised, transporttöölised), et neid edasi analüüsida ja hinnata sekkumismeetmete tulemuslikkust.

3 Veebirakenduse analüüs

3.1 Andmebaaasi valik

Andmebaas on rakenduse üks tähtsamaid osasid, kuna võimaldab suures mahus salvestada ning muuta rakendusega seotud andmeid. Andmebaasidel on turvalisuse tagamiseks erinevaid meetodeid, mis kindlustavad andmete turvalisust.

Andmebaasid jagunevad põhiliselt viieks erinevaks kategooriaks: relatsiooniline, mitterelatsiooniline, võrgumudeli põhine, objekt-orienteeritud ja hierarhisel mudelil põhinev andmebaas [8].

Relatsiooniline andmebaas ehk SQL andmebaas salvestab andmeid tabelites ning ridade kaupa. Selle tööpõhimõte seisneb informatsiooni sidumisel mitmetest tabelitest kokku, kasutades selleks võtmeväärtusi. Kõige populaarsemad nendest on Microsoft SQL Server, Oracle Database, MySQL ja IBM DB2 [9].

Mitterelatsiooniline andmebaas ehk NoSQL andmebaas salvestab andmeid kasutades hoidmise mudelit, mis on optimiseeritud spetsiaalsete nõuete jaoks tulenevalt andmestikust, mida salvestatakse. Enim kasutatavad mitterelatsioonilised andmebaasid on populaarsuse suhtes järjestatult: *MongoDB, Apache Cassandra, Redis, Couchbase* ja *Apache HBase* [9].

Võrgumudeli põhine andmebaas on sarnane hierarhilisel mudelil põhinevale andmebaasile, aga alam-olemid saavad ennast siduda mitme vanem-olemiga. Negatiivseks pooleks on selle struktuuri keerukus, mille tõttu võib osutuda väga raskeks seda muuta [8].

Objekt-orienteeritud andmebaasis on andmed esiatud objekti kujul. Kasutatakse suure jõudluse, keerukate algoritmide või kiiremate tulemuste puhul. Vähesed programmeerimise keeled toetavad objekt-orienteeritud andmebaase. Võrreldes teiste andmebaasidega on selle kasutamine tunduvalt keerkukam [8].

Hierarhilises andmebaasis on andmed puustruktuuriga, kus on alati kasutusel üks mitmele seosed. See ei ole paindlik olemite suhete poolest, mille tõttu ei ole see enam tänapäeval populaarne [8].

Käesoleva töö jaoks on kõige pädevam relatsiooniline andmebaas. Peamised põhjused valikuks on paindlikud suhted olemite vahel ja pikas perspektiivis andmemahu kasvades stabiilsuse ning töökindluse säilitamine.

Microsoft SQL Server on andmebaas, mis on arendatud ning avaldatud Microsoft-i poolt. Seda on kerge kasutada, ühtlasi on ka universaalne, kuna on võimalik kasutada Linuxi ja Windowsi operatsioonisüsteemide peal. Andmebaasi üle täielik kontroll puudub, kuna see on Microsofti halduses [10].

Oracle andmebaas, mis on arendatud Oracle'i korporatsiooni poolt. Lõputöö kirjutamise ajal on see kõige enamlevinud andmebaas [11]. Selle andmebaasi skeemiks on loend, mis koosneb loogilistest andmestruktuuridest. Seda on laiendatud objekt-orienteeritud andmebaasi mudelile, mis teeb selle keerukate ärimudelite salvestamise jaoks optimaalseks. See on ühtlasi üks põhjustest, miks Oracle andmebaas on nii kaua püsinud globaalses mastaabis populaarsuses esikohal [12]. Väiksema rakenduse korral on selle kasutusele võtmine kordades keerulisem, kui teiste enamkasutavate andmebaaside puhul [13].

MySQL on lõputöö kirjutamise ajal populaarseim avatud lähtekoodiga ja üksikisikutele tasuta. Seda on algajatel lihtne kasutada ning andmemahu voo kasvades suures mahus skaleeritav. Sellel on unikaalne salvestuse süsteem, mis teeb selle suure-jõuliseks ning andmetega seonduvad päringud efektiivseks. 2009 aastal omandas Oracle korporatsioon MySQL-i. Kommertsilikuks kasutuseks on litsents vajalik, kuna on Oracle omanduses [12].

MariaDB on relatsiooniline andmebaas, mis on loodud MySQL-i arendajate poolt. See on avatud lähtekoodiga ja seda on võimalik kasutada mitmete operatsioonisüsteemide peal. Andmebaasil on tugi JSON API-de jaoks [14].

PostgreSQL on avatud lähtekoodiga ja objekt-relatsioonilise süsteemiga andmebaasi haldamise süsteem. Kasvas välja California ülikooli projektist „Ingres" 1986 aastal. See on ühilduv mitmete platvormidega ning on toetatud optimiseeritud jõudluse funktsioonid, mis tavaliselt on saadaval ainult kallites kommertsilikes andmebaasides. Üldiselt on MySQL kiirem andmete töötluses kui PostgreSQL. Optimiseerida PostgreSQL-i on raskem kui MySQL -i, kuna see on suunatud eelkõige ühilduvusele [15].

Käesoleva rakenduse jaoks on valitud relatsioonilisest andmebaasidest MySQL. Peamine põhjus on töötegija kogemus MySQL andmebaasiga, lisaks on Eesti skoobis loodav rakendus optimaalne eelnevalt nimetatud andmebaasi suhtes, kuna tegemist ei ole suure andmemahu vooga rakendusega. Kasutatakse lisaks unikaalset süsteemi, mis teeb päringud efektiivseks.

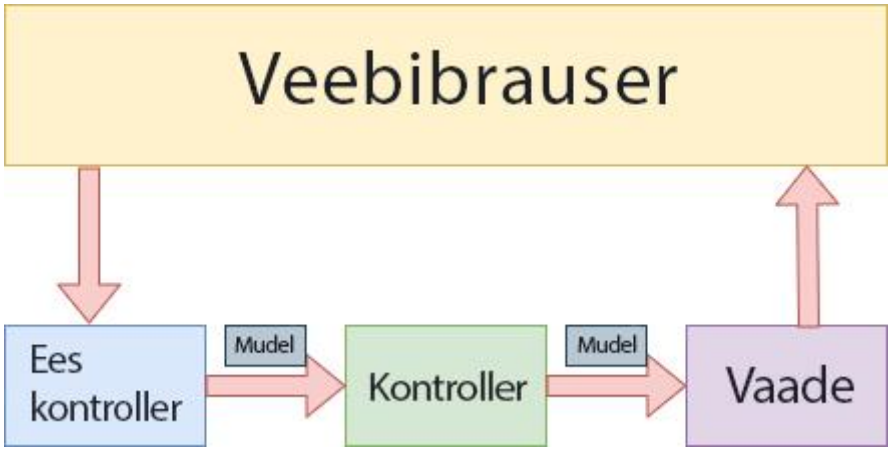
3.2 Tagarakenduse tehnoloogia valik

Tagarakenduse tehnoloogiate valik on mitmekesine, kuid kuna töötegija omab arvestatavat kogemust kolmes alljärgnevas programmeerimise keeles, siis tuleb valik teha nende vahel.

3.2.1 Tagarakenduse programmeerimiskeele valik

Java on programmeerimiskeel, mis on objektorienteeritud, klassipõhine ning millel on paralleelsuse tugi. Keel on arendatud Sun Microsystems poolt. See keel on loodud hajussüsteemides kasutamiseks [16].

Spring MVC on Java raamistik, mida kasutatakse veebirakenduste arendamiseks. Disainimustriks on MVC [16]. Joonisel 1 on välja toodud selle struktuur.



Joonis 1. Spring MVC Mudel.

Autori tõlge allikast [17].

C# on objektorienteeritud programmeerimiskeel. See keel on loodud Microsofti poolt ning töötab .NET raamistiku peal. Selle õppimise keerukus on tunduvalt lihtsam, kui Javal. Kuna see on objektorienteeritud keel, siis sarnaselt Javale annab see programmidele selge struktuuri ja lubab koodi taaskasutust [18].

Python on kõrgetasemeline ning objektorienteeritud programmeerimiskeel. Kasutatakse palju inglisekeelseid märksõnu keele lihtsustamiseks. Väga levinud algajate seas, kuna seda peetakse üheks lihtsamaks programmeerimiskeeleks.

Eelpool kirjeldatud keeltest valis autor C# programmeerimiskeele, eelkõige kogemuse poolest keelega. C# on keelena kaasaegsem kui Java, saab kasutada geneerikuid, mis on suur eelis võrreldes Javaga. Lisaks on lõputöö autori töökohal kasutusel C#. Samuti võib Javat kasutades lõputöö praktilise osa käigus ajakulu kujuneda tunduvalt suuremaks kui C#, kuna kogemus ei ole nii suur ning keelena on Java keerukam.

3.2.2 Raamistiku valik

Võttes kasutusele C#, tuleb valida ka vastav raamistik selle jaoks. Raamistik hõlmab endas kõiki vajalikke komponente, rakenduse programmeerimise liideseid ja mooduleid, mida on arenduseks vaja. Kuna Mircrosoft on välja tulnud uue .NET Core versiooniga .NET 5.0, mis sisaldab .NET Framework-i ning .NET Core'i enamikke komponente, siis lõputöö kirjutamise ajal on kõige mõistlikum valida hiljuti välja tulnud .NET 5.0. Samuti on see märgatavalt kiirem kui .NET Core 3.1, mis on senimaani laialt kasutuses olnud [19].

3.2.3 Kasutajaliidese ühendus

Kasutades .NET 5.0 on võimalik valida kahe lahenduse vahel, mille läbi andmed lõppkasutajale jõuavad.

- MVC-d kasutatakse, et luua veebirakendusi, mis tagastavad nii andmed kui vaate ühes. Kasutades seda mustrit suunatakse sissetulevad päringud kontrolleri, mis omakorda kasutab käesolevat mudelit, et täita kasutaja päringud. Pärast päringu täitmist valib kontroller vastava vaate, mida kuvada ning koos sellega ka vastavad andmed. Sellega saab ehitada dünaamilisi vaateid veebirakenduses. Mustri kasutamine tagab puhta rakenduse loogika eralduse [20].

- Web API tagastab ainult andmed. Kasutades seda on vaja lisaks kasutajaliidest ehk eesrakendust, mis lõppkasutajale tagastab Web API-st tulnud andmed korrastatud kujul. Sissetulevad päringud jagatakse vastavalt HTTP meetodite järgi. See on kergekaalulise arhitektuuriga [21].

Rakenduse jaoks on Web API kõige sobilikum, kuna sellega on võimalik kasutada REST lõppunkte ka teiste teenuste jaoks, kui tulevikus vajadus tekib. Näiteks mõnel firmal on soov saada analüüsi eesmärgil pidevalt uusimaid andmeid oma töötajate kohta. Sel juhul saavad nad teha päringuid vastava lõpp-punkti vastu, ning integreerida selle enda süsteemi. Lisaks on eraldatuna ees-ja tagarakenduse turvalisus suurem.

3.3 Eesrakenduse raamistiku valik

Eesrakendus on lõppkasutajale nähtav ja interaktiivne osa kogu tervikust. Eesrakendus võtab vastu sisestava info ning saadab selle tagarakendusele töötlemiseks. Eesrakenduste raamistike valik on lõputöö kirjutamise ajal rohkearvuline ning igal raamistikul on omad positiivsed ning negatiivsed küljed.

3.3.1 Vue

Vue on avatud lähtekoodiga Javascripti raamistik kasutajaliidese ehitamise jaoks. Selle stabiilne versioon tuli välja 2019 aastal. See on kergekaaluline ning lihtsam kui Angular keerukuse poolest, kuid sarnaneb kasutuse poolest Angularile, kuna on selle pealt ehitatud. Seda kasutatakse peamiselt üheleheliste veebirakenduste ehitamiseks. Vuel pole DOM uuendamine alati sama kerge, kui konkurentidel ning selle korrektselt töölesaamine võtab rohkem aega ja põhjalikke teadmisi [22].

3.3.2 React

React on Facebooki poolt 2013. aastal loodud platvormist sõltumatu teek. Kuna see pole otseselt raamistik, vaid teek, siis selle üldine keerukus on väiksem kui Angularil ning arendajatel on rohkem vabadust valida tööriistad, mida parasjagu vaja läheb. Selle mobiilne lahendus, React Native, on lõputöö kirjutamise ajal üks juhtivamaid tehnoloogiaid Androidi ja IOSi rakenduste seas. See sobib mitmesuguste lahenduste jaoks, kuid enamasti valitakse see keeruka veebi arenduse ning Native rakenduste jaoks [22].

3.3.3 Angular

Angular on Google'i poolt arendatud raamistik. Võrreldes Reacti ning Vuega on see nendest kõige vanem. Angular kasutab Typescripti, mis on Javascripti super komplekt.

See raamistik on algusest peale koos paljude lisadega. See on mõeldud suurte projektide jaoks, millel on väga mahukas funktsionaalsus [22].

Käesoleva eesrakenduse jaoks on valitud React. Võrreldes Angulari ja Vuega on Reacti funktsionaalsus tunduvalt suurem, kuna tegemist on teegiga. Kasutades Virtual DOM'-i on võimalik jälgida iga komponenti väärtusi eraldi ning vastavalt sellele ka neid uuendada [23]. Lisaks on rakenduse kasvades Reactile teste lihtne kirjutada tänu olemasolevatele tööriistadele. See on hea moodus rakenduse korrektse töökorra säilitamiseks pärast funktsionaalsuse refaktooringut või uue lisamisel [24].

3.4 Arenduskeskkonna valik

Arenduskeskkond on keskkond, kus on kasutatav kogu vajalik funktsionaalsus tarkvarasüsteemi arendamiseks. Käesoleva töö puhul käsitletakse selle all versioonihaldustarkvara ja arenduskeskkonda.

3.4.1 Versioonihaldustarkvarad

Versioonihaldustarkvaradest on võrdluseks toodud kolm kõige populaarsemat koodihalduse keskkonda, mis on tasuta kasutamiseks. Seejärel on välja toodud käesoleva töö puhul kõige sobilikum keskkond.

Gitlab on koodihalduskeskkond, mille asutasid Dmitriy Zaporozhets ja Valery Sizov 2011. aastal. Selles antakse täielik kontroll repositooriumite ja projektide üle ning on võimalik neid privaatseks või avalikuks teha. Funktsionaalsuse poolest on see sarnane GitHubiga, kuid repositooriumite allalaadimise ning üleslaadimise kiiruste poolest jääb see nimetatule alla [25].

GitHub asutati 2008. aastal San Franciscos Chris Wanstrathi, P. J. Hyetti, Tom Preston-Werneri, ja Scott Chaconi poolt. See on koodihalduskeskkond, mida peamiselt kasutatakse avatud lähtekoodiga projektide hoidmiseks. 2020. aasta alguses ületas selle kasutajate arv 40 miljonit. Praegu on Githubis võimalik luua repositooriume lõpmatu arv ning samuti kaasata lõpmatu arv kaaskasutajaid. Ühe repositooriumi suurus võib olla, kuni 500MB tasuta versioonis, mis võib tekitada probleeme suuremate projektide puhul [25].

BitBucket on koodihalduskeskkond, mis on osa Atlassiani tarkvarakomplektist. Seda on võimalik kasutada integreeritud vigade jälgimissüsteemiga JIRA. Sarnaselt GitHubile saab selles luua lõpmatu arv repositooriume. Algajate jaoks on see eelistatud koodihalduskeskkond tänu lihtsale kasutajaliidesele ning tõmbetaotlustele [26].

Käesoleva veebirakenduse jaoks on koodihalduskeskkonnana kasutusele võetud GitHub. Lõputöö autor kasutab igapäevaselt seda, teiste keskkondade kasutusele võtmine võtaks lisa aega ning nõuaks uue keskkonna juurde õppimist. Lõputöö kirjutamise ajal on see kõige enam kasutatud koodihalduskeskkond, mis tagab suure kogukonna toe.

3.4.2 Integreeritud arenduskeskkonna valik

Arenduskeskkondade valik tuleb teha eesrakenduse ning tagarakenduse jaoks. Allpool on välja toodud esmalt tagarakenduse ning siis eesrakenduse potentsiaalsed arenduskeskkonnad. Seejärel on välja toodud kõige sobilikumad arenduskeskkonnad ning nende põhjendused.

Tagarakenduse jaoks, mis kasutab .NETi on integreeritud arenduskeskkondadest kõige enam kasutatavad: Visual Studio, Visual Studio Code ja JetBrains Rider.

- Visual Studio Code on üks enam kasutavatest IDE'-dest, kuna seda on võimalik tasuta kasutada ning sellega on võimalik arendada mitmetes keeltes. Algselt on see kergekaaluline ning väga efektiivne, kuna see baseerub robustsel pluginate süsteemil, millega saab enda töökeskkonda üles seada. .NETi arenduseks on vaja lisada C# laiend.

- Visual Studio on täisfunktsionaalne integreeritud arenduskeskkond, mis on loodud Microsofti poolt ning seda saab tasuta kasutada. See on saadaval ainult Windows ja Mac-i operatsioonisüsteemide peal [27]. See pakub rohkem valikuvõimalusi, kui Visual Studio Code, aga selle arvelt tarbitakse süsteemi ressursse tunduvalt rohkem, mis muudab suurte projektidega töötamise raskendatuks.

- Rider on JetBrains-i poolt 2016. aastal loodud IDE, millel on ainult tasuline versioon. Seda on võimalik kasutada Windows, Mac ja Linuxi peal. Kui Visual Studioga funktsionaalsus muutub vastavalt opertasioonisüsteemile, siis Rideril jääb see igalpool samaks. Rideril on staatiline koodi analüüs, mis on täielikum,

kui Visual Studiol. See võimaldab automaatselt näha analoogseid, kuid optimiseeritud koodi blokke [28].

Kuna lõputöö autoril on ligipääs Rideri tasulisele IDE-le läbi ülikooli ning ta on harjunud seda kasutama, siis kõige sobilikum on teha seda ka käesoleva töö puhul. Võrreldes Visual Studioga on koodi analüüs tunduvalt parem ning kasulikum, mis aitab kaasa töö tõhususele. Rideri kasutajaliides on selgemini arusaadavam ning võimaldab efektiivsemat rakenduse arendust.

Eesrakenduse jaoks, mis on Reacti raamistikus, on alljärgnevalt välja toodud kõige populaarsemad arenduskeskkonnad, millega autoril on juba ulatuslik kogemus olemas.

- Webstorm IDE on arendatud ja hooldatud JetBrains poolt ning on avatud lähtekoodiga. See on saadaval Windowsi, Mac-i ja Linuxi operatsioonisüsteemides. Sellel on kohe algusest olemas kõik vajalik JavaScripti arenduseks, vastupidiselt Visual Studio Code-ile, kus peab laiendeid kasutama, et saada sama arenduskeskkond nagu WebStorm. Giti integratsioon on Webstormil täielikumalt välja arendatud, näiteks on võimalik vaadata koodi bloki ajalugu [29].

- Visual Studio Code-i on võimalik kohe kasutada Javascripti arenduseks ilma lisalaiendeid kasutamata.

Eesrakenduse jaoks on kõige mõistlikum valida WebStorm IDE, kuna see toetab produktiivsemat arenduse protsessi läbi erinevate integreeritud abiliste. Lisaks on lõputöö autoril läbi Ülikooli tasulise WebStormi versiooni kasutusvõimalus.

4 Arendus

Veebirakenduse arendus on jaotatud taga-ja eesrakenduse peatükkideks.

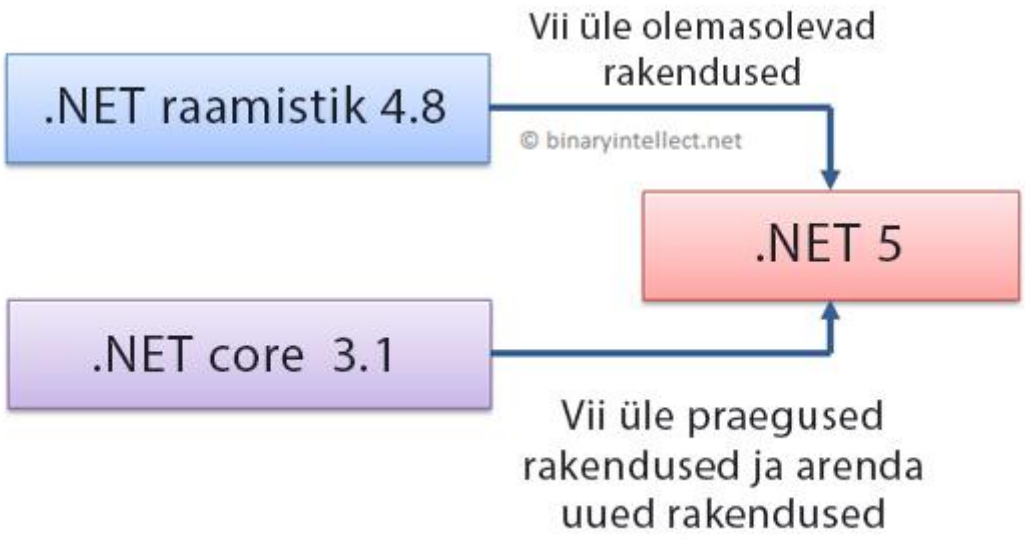
4.1 Tagarakenduse arendus

Tagarakendus kasutab REST API-t sissetulevatele päringutele vastamiseks. Samuti toimub andmete töötlemine ning äriloogika realiseerimine. Alljärgnevates peatükkides on lähemalt käsitletud kasutatud tehnoloogiaid ja arenduse erinevaid aspekte.

4.1.1 ASP.NET Core

ASP.NET Core on uus versioon ASP.NET Web Frameworkist, mis on mõeldud töötama .NET Core platvormi peal. See on tasuta, avatud lähtekoodiga, suure jõudlusega ja platvormi-ülene raamistik. Võimalik on majutada mitmel platvormil ning võrreldes .NET Framework-iga pole see IIS-st sõltuvuses [30].

Joonisel 2 on välja toodud NET Core 3.1 ning .NET raamistik 4.8 ühenduvus ühte raamistikku.



Joonis 2. .NET 5 raamistiku mudel.

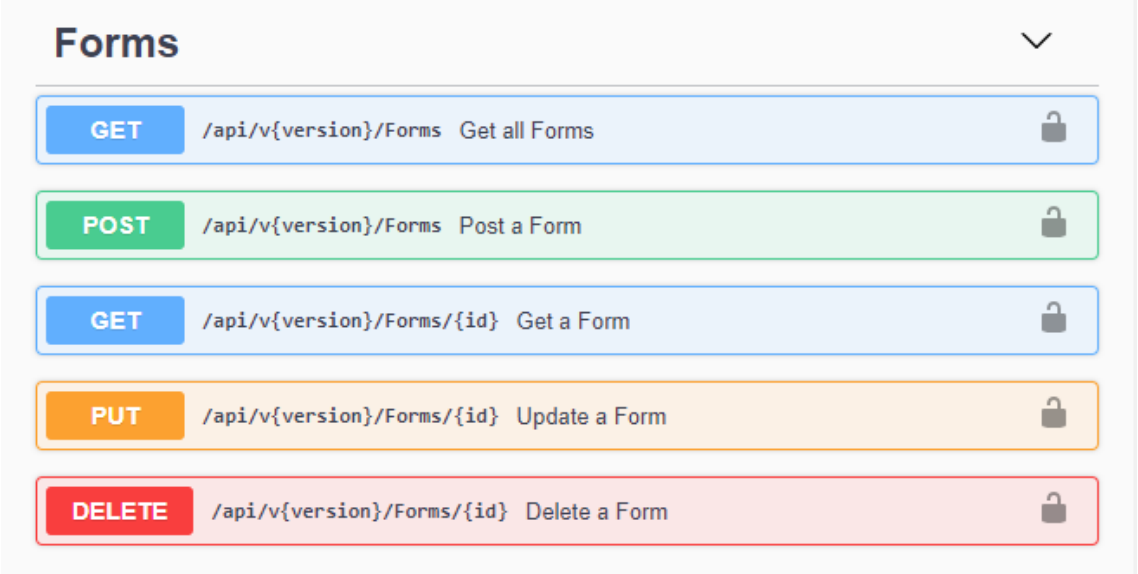
Autori tõlge allikast [31].

4.1.2 Tagarakenduse ülesehitus

Tagarakenduse projektid ehk klassi teegid baseeruvad .NET 5.0 peal. Arhitektuur jaotub neljaks põhiosaks. Kasutusel oleva arhitektuuri eesmärk on võimaldada rakendust tulevikus edasi arendada suuremas skoobis ilma suuremaid muutusi arhitektuuris tegemata. Alljärgnevalt on eraldi välja toodud kõik osad ning nende ülesanded ja kirjeldused.

- WebApp – Asuvad API kontrollerid, mille kaudu käib suhtlemine eesrakendusega. API on ülesehitatud häid tavasid järgides. Tagastatakse JSONi vormindatud sõne koos vastava sisutüübi päisega. Ressursside päringute nimed on alati mitmuses, et oleks võimalikult ühtne päringute struktuur. Vastusega lisaks tagastatakse vastav vastuse kood.

  Alljärgneval joonisel 3 on näha tagarakenduse ühe kontrolleri lõpp-punktide koos *CRUD* operatasioonidega, mis on genereeritud *Swagger*-i poolt.



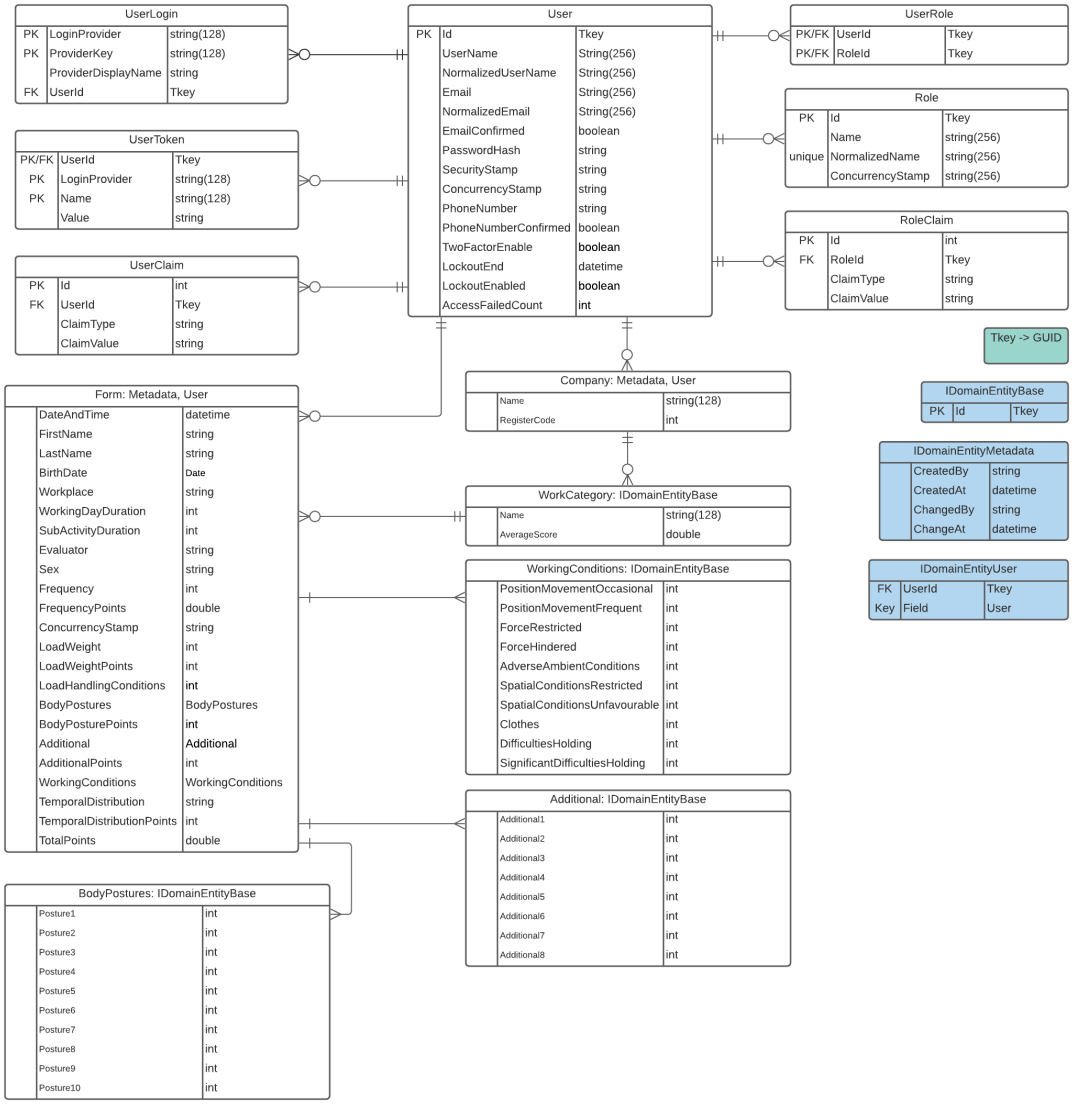
Joonis 3. Vormi kontrolleri lõpp-punktid

- Äriloogika kiht – see on rakenduses ühendavaks sillaks WebApp-i ja andmete juurdepääsu kihi vahel. Kõik sissetulevad andmed saadetakse läbi selle kihi, kus toimub äriloogika rakendus vastavalt andmetele. See on kõige tähtsam kiht arhitektuuris, kuna hõlmab endas kogu rakenduse äriloogikat.

- Andmete ligipääsu kiht – loob päringu äriloogika kihilt saadud parameetrite alusel. Toimub suhtlemine andmebaasiga, salvestab ning küsib andmeid andmebaasist.

4.1.3 Andmebaasi kavandamine

Enne andmebaasi teostust peab olema põhjalikult planeeritud andmebaasi struktuur ning sellega kaasnevad olemid, et tulevikus saaks võimalikult hõlpsasti muudatusi teha. Andmebaasi mudeli koostamiseks on kasutatud ERD metoodikat. Joonisel 4 on näha käesoleva rakenduse andmebaasi olemi-suhte diagramm.



Joonis 4. Andmebaasi olemi-suhte diagramm.

4.2 Eesrakenduse arendus

Eesrakendus on kontaktpunktiks lõppkasutajale, sissetulevad andmed saadetakse tagarakendusele ning sissetulevad andmed kuvatakse loetavas kontekstis. Arendus toimub Reacti raamistikuga ning WebStorm IDE keskkonnas. Alljärgnevates peatükkides on lähemalt käsitletud kasutatud tehnoloogiaid ja arenduse erinevaid aspekte.

4.2.1 ReactJS teek

React-iga tehtud rakendus on üles ehitatud mitmetest komponentidest, kus iga komponent tagastab tüki HTML koodi. Neid on võimalik kasutada mitmetes kohtades ja teiste komponentide sees, et luua mitmetasandiline rakendus. ReactJS kasutab virtuaalset DOM-il põhinevat mehhanismi, et näidata korrektset HTML DOM-i. Virtuaalne DOM muudab ainult individuaalseid elemente, mitte kogu DOM-i.

4.2.2 Eesrakenduse ülesehitus

Kogu rakendus on Typescript-is kirjutatud ning kasutab ES6-te. Arhitektuur koosneb viiest põhiosast. Alljärgnevalt on eraldi välja toodud kõik osad ning nende ülesanded ja kirjeldused:

- Komponentide kaust – sisaldab eraldiseisvaid UI komponente, mida on võimalik kasutada üle kogu rakenduse. Võimalik on taaskasutada mitmes kohas üldiseid komponente, et vähendada kogukoodi mahtu.

- Teenuste kaust – ei ole osa komponentidest, vaid API integratsioon, mille eesmärk on siduda rakendust tagarakendusega. Päringud tehakse kasutades Axios HTTP klienti. Turvalise suhtluse tagamiseks on kasutusel JWT, mille hoidmine käib rakenduse kontekstis.

- Konteksti kaust – annab võimaluse hoida globaalselt andmeid, ilma neid vanemkomponentidelt lapskomponentidele saatmata. Rakenduse skoobis hoitakse kasutajanime ja JWT-d.

- Hoidla kaust – hoitakse pilte ning css-i kaustasid, mida kasutatakse üle rakenduse.

- Utiliidi kaust – hoitakse väiksemaid funktsioone, mida kasutatakse korduvalt.

4.2.3 Rakenduse disain

Rakenduse disainimiseks kasutati Bootstrap-i, mis on tasuta ja avatud lähtekoodiga tööriistade kogum dünaamiliste veebilehtede loomiseks. See lihtsustab eesrakenduse disainimist, kuna ei pea kõike otsast lõpuni tegema, vaid on võimalik kasutada Bootstrapi poolt pakutavaid malle.

4.3 Rakenduse turvalisus

Tagarakenduse turvalisus põhineb ASP.NET Core turvalisustehnoloogiatel, mis pakub kaitset levinud rünnakute eest ning samuti kasutajatuvastust ja autoriseerimist. Autoriseerimine põhineb JWT standardi alusel (HS512 allkirja algoritmiga). Standardiga loodud märgendeid jagatakse ees ja-tagarakenduse vahel kasutades hetkel HTTP päringuid. Eesrakenduses hoitakse märgendeid Reacti rakenduse siseses mälus, mitte localStorage-is, kust on võimalik märgend kergesti kätte saada. JWT tehnoloogia kasutamine aitab ka CSRF rünnakute vastu.

SQL süstimise vastu on kasutusele võetud LINQ tehnoloogia, mille kaudu kõik päringud andmebaasi tehakse. Kasutades LINQ tehnoloogiat edastatakse andmed andmebaasi ainult parameetrite järgi. Selle abil saab vältida sõnede manipuleerimist ning nende kokku liitmist.

4.4 Valminud veebirakenduse analüüs

Töö käigus valmis veebirakendus, mis jaotub kolmeks osaks. Tavakasutajale ehk töölistele, kes tegelevad raskuste teisaldamisega, on mõeldud vormi täitmise vaade, kus on võimalik vormi korrektsel täitmisel saada tulemus oma töökeskkonna seisundi kohta. Vastavalt rakenduse poolt arvutatud skoorile kuvatakse vormi täitjale vastav info, mis annab ülevaate töökeskkonna ohutuse tasemest ning vajalikud juhised edasiseks tegutsemiseks.

Järgmine vaade eeldab, et kasutaja on sisselogitud. See on mõeldud ergonoomidele, töökeskkonnaspetsialistidele, kes hindavad töökeskkondi suuremas mastaabis ning kellele on vajalik täidetud vormide tulemuste salvestamine ning kategoriseerimine nii firmade kui ka individuaalsete töötajate tasemel. Vormide täitmine ning haldamine jaotub kahte kategooria: ettevõtted ja individuaalsed töölised. Ettevõtete jaoks on eraldi

sektsioon, kus on võimalik lisada uusi ettevõtteid ning hallata juba teostatud vorme ettevõtete kohta. Vormide lisamine ettevõttele käib läbi alamtöö kategooriate, mille kaudu vormid jagatakse vastava alamtöö kaupa ettevõttes. Teine kategooria on mõeldud individuaalsete töötajate jaoks. Sisestatakse vastav info kelle kohta vorm täidetakse ning täidetud vormid salvestatakse. Sarnaselt ettevõtete vormide haldamisele toimub ka individuaalsete töötajate vormide haldamine.

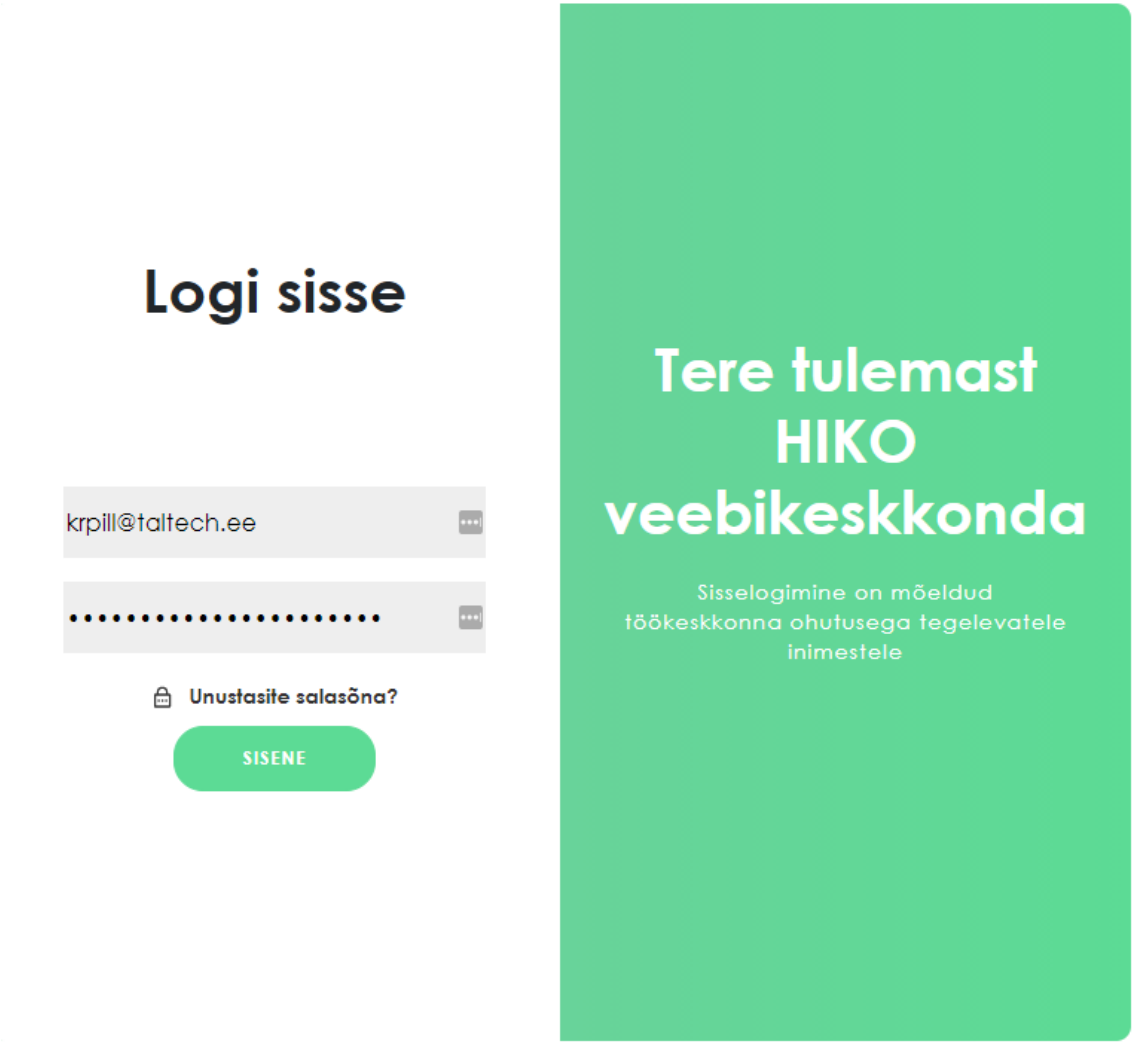
Viimane vaade on mõeldud rakenduse haldajale. Haldajal on võimalik tavakasutajate kontosid lisada ning vajadusel neid eemaldada. Samuti näeb üldinfot täidetud vormide ning kasutajate kohta.

5 Rakenduse realisatsioon

Käesolevas peatükis on välja toodud rakenduse osad ning nende funktsionaalsus, et anda ülevaade rakenduse struktuurist.

5.1 Kasutajatuvastus

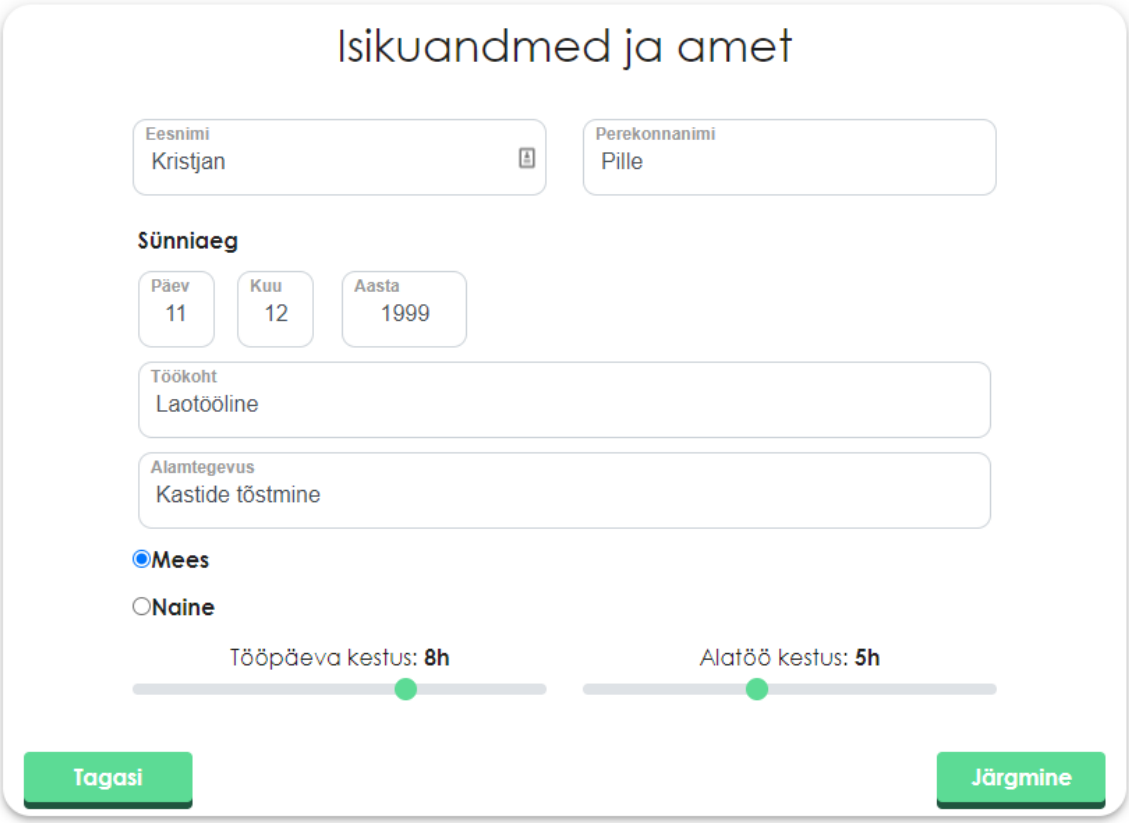
Sisselogimine on mõeldud ergonoomidele, töökeskkonnaspetsialistidele, ettevõtte omanikele või isikutele, kellel on vajadus vormide täitmisel neid ka salvestada ning saadud tulemusi edasi analüüsida. Sisselogitud kasutajad jagunevad õiguste poolest kaheks: tavaõigusega kasutaja ja rakenduse haldaja õigusega kasutaja. Alljärgneval joonisel 5 on välja toodud HIKO rakenduse sisselogimise vaade.



Joonis 5. Rakenduse sisselogimisleht.

5.2 Vormi täitmine

Vormi täitmisel on kaks erinevat võimalust. Kasutajad, kes sisselogima ei pea ehk töölised ja individuaalid, saavad vormi ära täita sisestades ainult minimaalse vajaliku informatsiooni. Sellisel juhul informatsiooni ei salvestata vormi täitmisel, kogu protsess toimub anonüümselt. Sisselogitud kasutajad, kes täidavad vorme töötajate kohta, peavad sisestama vastava informatsiooni, et hiljem vorme korrektselt identifitseerida oleks võimalik.



Joonis 6. Vormi algus.

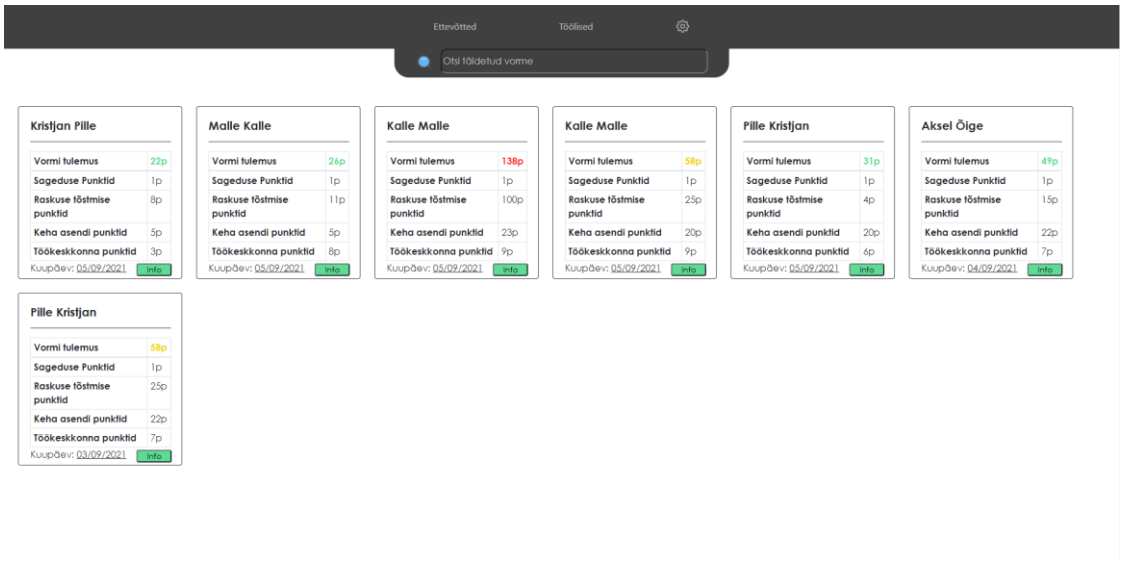
5.3 Vormi täitmise tulemused

Täidetud vormi tulemuse saab kasutaja koheselt teada ning vastavalt tulemusele on võimalik edasi tegutseda, et tagada tervist mitte ohustav töökeskkond. Igale töötajale kuvab veebirakendus numbrilise tulemuse terviseriski, teabe väsimuse võimaliku kujunemise ja meetmete kohta, mida saaks rakendada töötingimuste parandamiseks. Alljärgneval joonisel 7 on välja toodud töötaja individuaalne tulemus skooriga 29.

Joonis 7. Tulemuse vaade.

5.4 Salvestatud tulemused

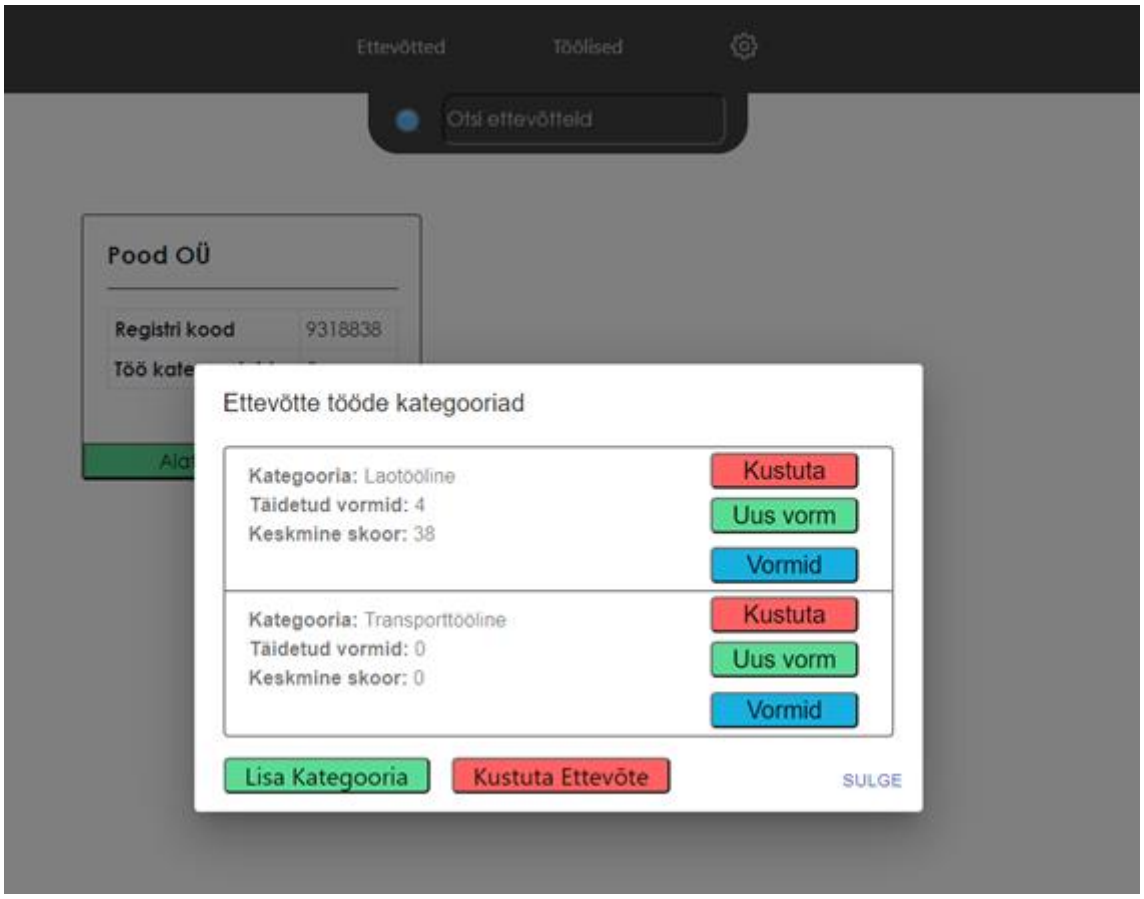
Sisselogitud kasutaja vormid salvestatakse automaatselt nende lõpetamisel. Täidetud vormid on välja toodud eraldi vaates. Lisaks saab iga individuaalse vormi kohta ka täpsemini vaadata, mis kategoorias punktid kõige kõrgemad olid ning vastavalt edasi tegutseda. Joonisel 8 on kuvatud näide juba tehtud vormidest.



Joonis 8. Töötajate vormid.

5.5 Ettevõtted

Ettevõtete vaade on ülesehituselt sarnane töötajate vormide vaatele. Avades individuaalse ettevõtte kuvatakse kasutajale töötajate kategooriad. Järgnevalt on võimalik töötajate kategooriaid lisada ning neid eemaldada. Töökategooriate alla on võimalik lisada uusi vorme ning näha seniseid täidetud vormide tulemusi. Alljärgneval joonisel 9 on välja toodud vaade ühe ettevõte kohta.



Joonis 9. Ettevõtte põhine vaade.

5.6 Vormi valideerimine

Vormi käsitsi täitmisel on võimalik, et vormi tulemus ei ole korrektne, kuna vorm on valesti täidetud. Näiteks on mõni punkt vahele jäetud või valiti kategooriatest liialdaselt valikuid. Käesolevas rakenduses on sisseehitatud valideerimise süsteem, mis tagab alati korrektse vormi täitmise ja tulemuse. Joonisel 10 on näide sellest, kui kasutaja on vormil unustanud valida efektiivse raskuse kaalu.

Joonis 10. Vormi valideerimise näide.

5.7 Kasutajate haldus

Rakenduses on eraldi sektsioon rakenduse haldajale. Selles sektsioonis on võimalik hallata registreerunud kasutajaid. Lisaks saab näha kõikide täidetud vormide keskmist tulemust ning veel täiendavat informatsiooni, kuid vormidega ei tule kaasa vormi kohta käiva isiku andmeid.

6 Tulevikuarendused

Töö tulemusena on valminud täisfunktsionaalne veebirakendus. Arenduses on suur osa rõhku pandud veebirakenduse nutiseadmete kasutuskogemusele. Eesrakendus on hetkel eestikeelne. Tagarakendusele on lisatud i18n keelesüsteem, lahendatud süsteemiga hoiustatakse tõlkeid andmebaasis ning vastavalt keele valikule tehakse vastav päring andmebaasi.

Hetkel kuvatakse vormi täitmisel tulemus ainult veebirakenduse siseselt, kuid lähitulevikus on plaanis lisada rakendusele funktsionaalsus, mis võimaldab soovi korral vormi täitjal saada tulemus e-posti aadressile.

Arenduse käigus kujundatud tagarakenduse struktuur võimaldab hõlpsasti lisada uusi ergonoomika hindamise vorme lisaks praegusele digitaliseeritud vormile. Erineva iseloomuga füüsilise töö koormuse hindamiseks on võimalik sel juhul valida ergonoomil või töökeskkonnaspetsialistil sobivaim vorm (nt korduvatest käte tööliigutustest tuleneva koormuse hindamine).

Kokkuvõte

Bakalaureusetöö tulemusena on loodud veebirakendus, mis võimaldab töökeskkondades hinnata raskuste käsitsi teisaldamise terviseriske. Rakendus põhineb innovatiivsel Saksamaa Riikliku Töötervishoiu Instituudi Baua poolt välja arendatud metoodikal. See on esimene raskuste käsitsi teisaldamise hindamise metoodika, mis haarab võimalikult palju teisaldamisega seotud töö iseloomu erinevaid aspekte ja ei ole piiratud ainult staatilise tööga.

Rakendust saavad kasutada nii erialase väljaõppega ergonoomid, töökeskkonnaspetsialistid kui ka tavatöötajad. Veebirakendus on taolisele töövahendile Eestis esmakordne. Andmete salvestamine loob võimaluse edaspidiseks andmeanalüüsiks, andmete taasesitamiseks, sekkumismeetmete hindamiseks dünaamikas ning elektroonseks andmete säilitamiseks. Digitaalne rakendus on mugav ja kiire kasutada ning annab koheselt tulemuste skoori erinevalt pabervormil käsitsi kokku arvutamisest.

Lihtne ja kättesaadav metoodika võimaldab lisaks tööandjate vastutusele ka töötajatel endil pöörata tähelepanu, et töökeskkond ei oleks tervist kahjustav.

Kasutatud kirjandus

[1] M. Douwes, „https://oshwiki.eu,“ 22 September 2020. [Võrgumaterjal]. Available: https://oshwiki.eu/wiki/Lifting_operations_and_lifting_equipment. [Kasutatud 26 February 2021].

[2] Riigiteataja, „https://www.riigiteataja.ee/,“ Riigiteataja, [Võrgumaterjal]. Available: https://www.riigiteataja.ee/akt/84808?leiaKehtiv. [Kasutatud 10 May 2021].

[3] G. Ahonen, „oshwiki,“ 24 April 2013. [Võrgumaterjal]. Available: https://oshwiki.eu/wiki/The_economic_dimension_of_occupational_safety_and_health_management. [Kasutatud 27 February 2021].

[4] Baua Federal Institute for Occupational safety and health, „Key indicator method: Baua,“ Baua, 1 June 2019. [Võrgumaterjal]. Available: https://www.baua.de/EN/Topics/Work-design/Physical-workload/Key-indicator-method/Key-indicator-method_node.html. [Kasutatud 27 February 2021].

[5] Tööinspektsioon, „tooelu,“ Tööinspektsioon, 6 May 2016. [Võrgumaterjal]. Available: https://www.tooelu.ee/et/Tooandjale/Tookeskkond/Tookeskkonna-korraldus/riskianalyys-riskide-hindamine-ja-ohjamine/riski-suuruse-hindamine. [Kasutatud 25 April 2021].

[6] W. S. Marras, „Fundamentals and assessment tools for occupational ergonomics,“ %1 *Fundamentals and assessment tools for occupational ergonomics,* Columbus, Waldemar Karwowski, 2006, p. 1024.

[7] E. Plus, „https://ergo-plus.com/,“ Ergo Plus, [Võrgumaterjal]. Available: https://ergo-plus.com/. [Kasutatud 11 05 2021].

[8] O. G. Limited, „Rula,“ Osmond Group Limited , [Võrgumaterjal]. Available: https://www.rula.co.uk/index.html. [Kasutatud 11 May 2021].

[9] S. Baroth, „Geeksforgeeks,“ Geeksforgeeks, 12 November 2020. [Võrgumaterjal]. Available: https://www.geeksforgeeks.org/types-of-databases/. [Kasutatud 6 March 2021].

[10] T. Pattinson, „pluralsight,“ pluralsight inc, 13 August 2020. [Võrgumaterjal]. Available: https://www.pluralsight.com/blog/software-development/relational-vs-non-relational-databases. [Kasutatud 6 March 2021].

[11] S. Jenny, „Geeksforgeeks,“ Geeksforgeeks, 10 April 2020. [Võrgumaterjal]. Available: https://www.geeksforgeeks.org/advantages-and-disadvantages-of-sql/. [Kasutatud 06 03 2021].

[12] S. IT, „db-engines,“ IT gmbh, 1 March 2021. [Võrgumaterjal]. Available: https://db-engines.com/en/ranking. [Kasutatud 7 March 2021].

[13] M. Chand, „C-sharpcorner,“ C-sharpcorner, 13 Julu 2019. [Võrgumaterjal]. Available: https://www.c-sharpcorner.com/article/what-are-the-most-popular-relational-databases/. [Kasutatud 07 March 2021].

[14] D. Sullivan, „Techwalla,“ Techwalla, 1 April 2015. [Võrgumaterjal]. Available: https://www.techwalla.com/articles/the-advantages-of-db2. [Kasutatud 7 March 2021].

[15] A. Morozov, „Mariadb,“ 4 May 2015. [Võrgumaterjal]. Available: https://mariadb.org/about/. [Kasutatud 24 April 2021].

[16] Guru99, „Guru99,“ Guru99, 1 January 2020. [Võrgumaterjal]. Available: https://www.guru99.com/introduction-postgresql.html#1. [Kasutatud 7 March 2021].

[17] Codeinstitute, „Codeinstitute,“ Codeinstitute, 14 October 2014. [Võrgumaterjal]. Available: https://codeinstitute.net/blog/what-is-java/. [Kasutatud 12 03 2021].

[18] Javatpoint, „https://www.javatpoint.com,“ Javatpoint, 2018. [Võrgumaterjal]. Available: https://www.javatpoint.com/spring-mvc-tutorial. [Kasutatud 9 May 2021].

[19] Geeksforgeeks, „Geeksforgeeks,“ Geeksforgeeks, 19 April 2018. [Võrgumaterjal]. Available: https://www.geeksforgeeks.org/introduction-to-c-sharp/. [Kasutatud 12 March 2021].

[20] A. Yakunin, „Medium,“ Medium, 20 November 2020. [Võrgumaterjal]. Available: https://medium.com/swlh/astonishing-performance-of-net-5-more-data-5cdc8d821e8c. [Kasutatud 13 March 2021].

[21] S. Smith, „Microsoft,“ Microsoft, 2 December 2020. [Võrgumaterjal]. Available: https://docs.microsoft.com/en-us/aspnet/core/mvc/overview?view=aspnetcore-5.0. [Kasutatud 13 March 2021].

[22] S. Chauhan, „Dotnettricks,“ Dotnettricks, 2 April 2019. [Võrgumaterjal]. Available: https://www.dotnettricks.com/learn/webapi/difference-between-aspnet-mvc-and-aspnet-web-api. [Kasutatud 13 March 2021].

[23] P. Mikus, „Medium,“ Medium, 25 February 2019. [Võrgumaterjal]. Available: https://medium.com/js-dojo/angular-vs-react-vs-vue-choosing-the-front-end-framework-b24302818fb1. [Kasutatud 13 March 2021].

[24] S. Surve, „Freecodecamp,“ Freecodecamp, 18 February 2021. [Võrgumaterjal]. Available: https://www.freecodecamp.org/news/why-use-react-for-web-development/. [Kasutatud 13 March 2021].

[25] E. Elliot, „Medium,“ Medium, 7 March 2019. [Võrgumaterjal]. Available: https://medium.com/javascript-scene/unit-testing-react-components-aeda9a44aae2. [Kasutatud 13 March 2021].

[26] Momchil, „Hackernoon,“ Hackernoon, 17 April 2020. [Võrgumaterjal]. Available: https://hackernoon.com/gitlab-vs-github-repositories-ci-deployment-devops-pricing-documentation-and-more-analysis-12qc32n9. [Kasutatud 14 March 2021].

[27] A. T. Tunggal, „Upguard,“ Upguard, 15 February 2021. [Võrgumaterjal]. Available: https://www.upguard.com/blog/bitbucket-vs-github. [Kasutatud 14 March 2021].

[28] Sheldon, „Teckangaroo,“ Teckangaroo, 16 October 2020. [Võrgumaterjal]. Available: https://teckangaroo.com/visual-studio-vs-visual-studio-code-latest/. [Kasutatud 14 March 2021].

[29] R. Peres, „Stackify,“ Stackify, 12 March 2018. [Võrgumaterjal]. Available: https://stackify.com/visual-studio-rider/. [Kasutatud 14 March 2021].

[30] E. Boyle, „Dzone,“ Dzone, 26 May 2020. [Võrgumaterjal]. Available: https://dzone.com/articles/10-reasons-why-webstorm-is-better-than-vs-code. [Kasutatud 14 March 2012].

[31] M. J. Price, C# 8.0 and NET Core 3.0 Modern Cross-Platform Development, Birmingham: Packt Publishing Ltd, 2019.

[32] Binaryintellect, „http://www.binaryintellect.net,“ Binaryintellect, 23 March 2020. [Võrgumaterjal]. Available: http://www.binaryintellect.net/articles/567c6c06-a2aa-44da-a443-78dafedebe65.aspx. [Kasutatud 9 May 2021].

[33] P. Kumar, „Dotnettutorials,“ Dotnettutorials, 12 January 2019. [Võrgumaterjal]. Available: https://dotnettutorials.net/lesson/introduction-to-asp-net-core/. [Kasutatud 19 March 2021].

[34] F. Manca, „Florimond,“ Florimond, 26 August 2018. [Võrgumaterjal]. Available: https://florimond.dev/blog/articles/2018/08/restful-api-design-13-best-practices-to-make-your-users-happy/. [Kasutatud 19 March 2021].

[35] „https://www.javatpoint.com,“ Javatpoint, 2018. [Võrgumaterjal]. Available: https://www.javatpoint.com/spring-mvc-tutorial. [Kasutatud 9 May 2021].

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

Mina, Kristjan Pille

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose FÜÜSILISE TÖÖ MONITOORINGUSÜSTEEMI LOOMINE, mille juhendaja on Kaido Kikkas
   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.

17.05.2021

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.