TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Helen Lepiku 193856IADB

Veebirakendus individuaalseks õmblemiseks kehamõõtude järgi

Bakalaureusetöö

Juhendajad: Meelis Antoi
Magistrikraad
Andres Käver
Magistrikraad

Tallinn 2022

Autorideklaratsioon

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

Autor: Helen Lepiku

24.04.2022

Annotatsioon

Käesoleva lõputöö eesmärgiks on luua veebirakendus, mis arvutab vastavalt kasutaja kehakujule vajaminevad suurused. Väljaarvutatud suuruste abil saab kasutaja alla laadida faili ja välja lõigata vastavate suurustega lõike. Seejärel saab suunduda kasutaja juba selgitavate õpetuste juurde, mis toetavad kasutajat kogu õmblusprotsessi vältel.

Antud lõputöös leiab ülevaate sarnastest olemasolevatest rakendustest ning tuuakse välja nende rakenduste puudused, võrreldes loodava rakendusega. Samuti määratletakse tulevase rakenduse nõuded. Seejärel kirjeldatakse erinevate tehnoloogiate valikuid ja käesoleva rakenduse arenduskäiku.

Töö tulemusena valmib veebirakendus, mis toetab kasutajaid igal sammul, et valmis õmmelda valitud toode. Samuti on antud rakendus hea võimalus õppimiseks, kuna kõik etapid on detailselt lahti kirjutatud ja nende juurest võib leida abistavaid pildimaterjale. Kasutajatele avaneb ka ülevaade juba nende poolt valmisõmmeldud ja pooleliolevatest toodetest.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 34 leheküljel, 7 peatükki, 22 joonist, 2 tabelit.

Abstract

Web Application for Individual Sewing Based on Body Measurements

The aim of this thesis is to create a web-based application that calculates the user's clothing size according to their body shape. Based on the calculated clothing size, the user can then download and cut out a sewing pattern with the corresponding size. Later, the user is directed towards explanatory tutorials that will guide them throughout the sewing process.

This thesis gives an overview of similar existing applications and discusses their shortcomings compared to the application created in this thesis. Also describing the development process of the application, the choice of different technologies, and the application's requirements.

The thesis results in a web-based application that supports its users at every step of the way to sew their selected products. This application is a great way to learn as all the steps are described in great detail and accompanied by useful imagery. The users are also given an overview of the products they have already sewn or that are currently in progress.

The thesis is in Estonian and contains 34 pages of text, 7 chapters, 22 figures, 2 tables.

Lühendite ja mõistete sõnastik

API	*Application Programming Interface*, rakendusliides
Backend	Teenusepoolne rakendus
BLL	*Business logic layer,* äriloogika kiht
DAL	*Data access layer,* andmete juurdepääsu kiht
CAD	*Computer-aided design*, raalprojekteerimine
CPU	*Central processing unit*, protsessor
CRUD	*Create*, *Read, Update, Delete*, Rakenduse tüüp andmebaasiga suhtlemiseks, mis peab suutma andmeid luua, lugeda, muuta ja kustutada
CSS	*Cascading StyleSheets*, küljendamisel kasutatav märgistuskeel
DOM	*Document Object Model*, dokumentidega suhtlemise liides
EF	*Entity Framework*, objektide ja relatsioonide kaardistamiseks mõeldud raamistik
FormData	Võti-väärtus paaride edastamiseks
Frontend	Kliendipoolne rakendus
HTML	*Hypertext Markup Language*, märgenduskeel veebilehe struktuuri defineerimiseks
HTTP	*Hypertext Transfer Protocol*, hüperteksti edastusprotokoll
JSON	*JavaScript Object Notation,* andmevahetusformaat
JWT	*JSON Web Token*, standard andmete loomiseks koos digiallkirjaga ja/või krüpteerimiseks
LINQ	*Language Integrated Query*, .NET raamistiku komponent andmebaasist pärimiseks
MPA	*Multi-page Application*, mitmeleherakendus
REST	*Representational state transfer*, veebiteenusega suhtlemise liidese arhitektuur
SPA	*Single-page Application,* üheleherakendus
SQL	*Structured Query Language*, andmebaasi suhtluskeel
UI	*User Interface,* kasutajaliides
URL	*Uniform Resource Locator*, veebiaadress

XSS	*Cross-site Scripting*, kindlat tüüpi rünnakute kirjeldamiseks

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

Läbi aegade on inimesi huvitanud mood ja paljud on tahtnud moega kaasas käia. Praegusel ajal on kauplustes rõivaste valik üsnagi suur, küll aga sõltuvalt hetkelisest moesuunast on erinevates poodides tegumood ja kangad sarnased. Nii mõnigi kord ollakse silmitsi probleemiga, et soovitud eset ei leita ühestki poest või siis riided ei istu piisavalt hästi seljas. Parim lahendus sellises olukorras oleks minna õmbleja juurde ja tellida just enda kehakuju ja maitse järgi riided. Õmblustööd on kallid ja enamik inimesi seda endale lubada ei saa. Võimalus oleks ka riided ise õmmelda, kuid tihtilugu jääb inimestel oskustest puudu. Seega ei jäägi muud üle, kui osta endale riided poest.

Õmblemine nõuab üsna palju teadmisi. Kõigepealt tuleb alustada lõike konstrueerimisega ja alles seejärel saab alustada detailide väljalõikamise ja õmblemisega. Lõike konstrueerimine on ajakulukas ja nõuab palju teadmisi. Sellepärast valitaksegi enamasti juba valmis konstrueeritud lõiked, mis on tehtud tüüpmõõtude järgi. Valmis lõiked on saadaval interneti avarustes kui ka ajakirjades. Lõigete puhul, mis on võetud ajakirjadest või internetist, kaasnevad aga suured riskid. Nimelt on inimeste kehakujud erinevad ja valmis lõige ei pruugi kõigile sobida. Võib juhtuda, et valminud toode ei mahu selga. Kuna kanga hind on tihti üsna kallis, siis ei soovi inimesed olla probleemi ees, et valmisõmmeldud toode ei sobi selga. Mida siis teha, kui jääb puudu oskustest ja teadmistest, et lõige konstrueerida?

Käesoleva lõputöö eesmärgiks on luua veebileht, mille abiga suudab inimene õmmelda endale just enda kehakuju arvestades riideesemeid. Samuti ei eelda veebileht inimestelt erilisi kogemusi või oskusi. Seega on tulevane veebileht sobilik kasutamaks ka alles õmblemisega tegelema hakkavatele inimestele. Veebileht arvutab välja lõigete suurused, mis vähendab riski, et õmmeldav ese ei hakka seljas istuma. Samuti leiab veebilehelt samm-sammult õpetuse, kuidas toode valmis õmmelda.

2 Ülesande püstitus

Hetkel puudub inimestel, kes hobikorras õmblevad, võimalus vastavalt enda kehamõõtudele õmmelda riideid. Lõigete konstrueerimine on aga aeganõudev ja keeruline ning nii mõningad hobikorras õmblejad ei soovi sellega tegelda, soovides kasutada valmis lõikeid. Samuti puudub lõigete juures õpetus, kuidas toode valmis õmmelda ja tihti tuleb otsida internetist abi või katseeksituse meetodil ise valmis teha.

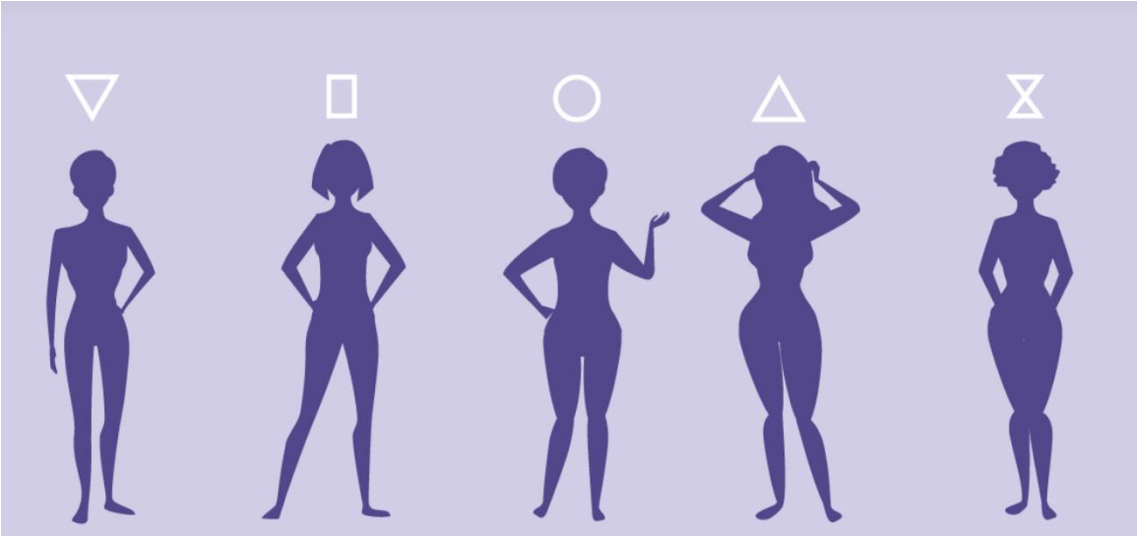
Antud diplomitöö autor pakubki välja lahenduseks veebirakenduse, milles kasutaja saab sisestada enda keha mõõdud. Käesolevas rakenduses saab kasutaja valida soovitud toote, mida ta sooviks endale õmmelda ja seejärel avaneb kasutajale võimalus lõige alla laadida. Kuna lõikel on kujutatud erinevad suurused, siis tagastab programm mõõtmed, mis sobivad kasutaja kehakujuga. Kasutaja tööks jääb antud pdf välja printida ja seejärel vastavalt enda mõõtudele lõigata lõige välja. Järgnevalt saab kasutaja edasi liikuda ja vaadata samm-sammult juhendit, mida tuleks teha, et toode lõpuni valmis õmmelda.

Lahendus käsitleb vaid tasuta versiooniga kasutajaid. Tulevikuplaanideks on soov teha võimalikuks kasutajal sisestada juba mitmeid erinevaid kehamõõtusid ühele tootele. Selline funktsionaalsus kuuluks juba tasulisse versiooni, kuna kasutaja arvatavasti on hakanud tegelema juba teistele inimestele toodete õmblemisega. Samuti tasulisest versioonist leiab väga keerukate esemete õmblemise, näiteks mantlite puhul, mille tegemine on juba väga keerukas ja nõuab väga detailset õpetust. Tehtav veebirakendus tuleb esialgu vaid eesti keeles. Hiljem edasi arendades on kindlasti soov lisada mitmeid keelevalikuid juurde, et kasutajaskonda suurendada.

3 Ülevaade probleemist

Peamiselt tuntakse viit erinevat kehatüüpi inimesi. Kehakujude illustreerivad joonised on leitavad Joonis 1.

Esimeseks on tagurpidi kolmnurk, mille puhul on inimesel kitsas vöökoht, õlad on laiemad kui puusad. Teiseks on ristkülik, mille puhul asuvad inimesel õlad, vöökoht ja puusad samal joonel. Ristküliku puhul ei ole eristatavat keskkohta. Kolmandaks on ring ehk teisisõnu ka õun. Õuna puhul on inimesel kõige laiemaks kohaks keskkoht. Neljandaks loetakse kolmnurka ehk pirni. Pirni puhul on kõige laiemaks kohaks inimesel puusad, õlad on tunduvalt kitsamad. Viiendaks kehakujuks peetakse liivakella. Liivakella puhul on puusad ja õlad sama laiad ja vöökoht tunduvalt peenem. Need kehakujud on kõige tüüpilisemad näited, aga sõltuvalt inimese treenitusest või toitumisharjumustest varieeruvad kehakujud veelgi rohkem. Paljud inimesed pole teadlikud enda kehakujust ja ei arvesta seda riiete valimisel. Tegelikult kõik lõiked ei olegi sobilikud iga kehatüübiga inimestele [1].



Joonis 1. Inimeste erinevad kehakujud

Õmblemise puhul on kõige olulisem samm konstrueerida lõige. Kui juba eos on protsess vale, siis lõpptulemus ei pruugi kujuneda selliseks nagu soovitud. Lõike konstrueerimine pole aga lihtne ja väga paljud hobikorras õmblejad ei soovi sellega tegelda ja valivad

valmislõike, mis on genereeritud tüüpmõõtude järgi. Kuna kehakujud on erinevad, siis kindlasti ei sobi kõik lõiked igaühele. Hobikorras õmbleja ei pruugi esmalt arugi saada, et vastav lõige ei ole õige suurusega. Õmblemisprotsess on väga pikk ja aeganõudev. Kui lõpptulemus on väga kaugel sellest, mida oodati, siis tihtilugu antakse alla ja visatakse ebaõnnestunud toode minema. See kõik reostab keskkonda ja on kahjulik inimese rahakotile.

3.1 Loetelu sarnastest lahendustest

Järgnevalt on välja toodud populaarsemad keskkonnad, mida kasutajad kasutavad kas lõigete konstrueerimisel või valmislõigete leidmiseks.

3.1.1 Modaris Lectra

Lectra pakub väga palju erinevaid programme oma klientidele. Programmide valik hõlmab kogu moedisaini erinevaid etappe: alustades kavandite visandamisest, edasi minnes tehnoloogiliste jooniste ja lõigete konstrueerimiseni. Lõigete konstrueerimisel ongi peamiseks Modaris programm. Samuti pakub Lectra programmi juurdelõikuseks, kus teostatakse kangast detailide väljalõikamist. Lisaks leiab Lectra programmide valikust kangakulu arvestamiseks vajamineva programmi. Tänu Lectrale on rõivaste valmimise protsess kiirenenud. Kuna Lectra programmide loetelust leiab kõik vajalikud programmid, siis tänu sellele on Lectra muutunud üheks kasutatavamaks moedisainerite seas [2]. Loetletud programmid on küll väga head aga neil puudub tasuta versioon. Peamiselt kasutavad Lectra programme suurettevõtted masstoodangu tootmisel. Programmi maksumus sõltub programmide valikust ja hinnapäringu jaoks tuleb võtta ühendust otse firmaga. Suurusjärgus võib programmide maksumus ulatuda mõnest tuhandest kuni kahekümne tuhandeni. Seega Lectra puhul üheks suurimaks miinuseks on kindlasti hind.

3.1.2 TukaCAD

TukaCAD on programm lõigete konstrueerimiseks. Lõikeid saab ise isikupärastada ja programm arvestab sisestatud mõõtudega. Programmi suureks plussiks on lihtsalt ja kiirelt mõõtude muutmisel saadav tulemus. TukaCAD-i kasutamisel on suureks toeks õppevideod, mille on firma ise loonud. Õppevideod hõlmavad peamisi operatsioone programmis.

Programm ei sobi kindlasti algajatele ja kasutaja peaks tundma vähemalt elementaarseid teadmisi lõike konstrueerimisest, kuna programm ei kontrolli lõike reaalset vastavust inimese kehaga. Seega on kasutajal võimalus genereerida lõikeid, mis sobivad hiljem kangast väljalõikamiseks. Samuti on antud programm tasuline. Algajate versiooni maksumus on 19 dollarit kuus, mis on üle seitsmeteistkümne euro. Algajate versioonis on palju asju piiratud ja populaarsem versioon on tunduvalt kallim ehk 99 dollarit kuus, mis on üle 89 euro [3]. Kohe kindlasti ei sobi antud lahendus hobiõmblejatele, sest maksumus oleks liiga kallis. Lisaks ei pruugi hobiõmblejal olla piisavalt teadmisi lõike konstrueerimisest.

3.1.3 Seamly 2D

Seamly 2D on avatud lähtekoodiga lõigete konstrueerimise tarkvara. Suureks plussiks on kindlasti see, et programm on tasuta ja saadav kõigile. Erinevad operatsioonisüsteemid toetavad programmi kasutamist [4]. Programmi saab sisestada enda mõõdud ja vastavalt nendele konstrueerida endale lõikeid. Võrreldes eelmise programmiga puudub sel programmil nii suur tugi ja on vähem funktsionaalsust.

Nii nagu ka eelmise programmi puhul, siis Seamly 2D ei sobi algajatele, sest kasutajal peaks olemas olema baasteadmised lõike konstrueerimisest. Lisaks võtab palju aega programmiga tutvumine ja kuna juhendite hulk on üsna limiteeritud, tuleb palju ise uurida ja vaeva näha.

3.1.4 Burda

Burda puhul on tegemist ajakirjaga. Lisaks on neil oma veebileht, kust on võimalik kasutajal osta lõige. Lõike maksumus sõltub keerukuse astmest, kuid ühe lõike hind on varieeruvalt 6 - 12 eurot. Lõiked on koostatud tüüpmõõtude järgi. Burdat on võimalik laenutada ka näiteks raamatukogust ja siis ei ole vaja kasutajal lõike eest maksta. Ajakirjas on leitav ka õpetus, mis ei ole väga detailne, kuid on sobilik juba õmblemisega tegelevatele inimestele.

3.1.5 Programmide kokkuvõte

Hetkel pakuvad rakendused lõigete konstrueerimist eeldusel, et kasutajal on eelnevad teadmised. Ükski programm ei kontrolli lõike õigsust, seega võib inimene oma loominguga konstrueerida ükskõik millise lõike ja tulemus võib olla mitte soovitud.

Samuti puudub väljatoodud programmidel juhendid edasiseks õmblusprotsessiks. Kuna iga toote puhul on erinevad etapid, siis ei pruugi kasutaja teada, millises järjekorras peaks protsessi läbima. Rakenduste võrdlused ja nende hinnaklass on toodud välja Tabel 1.

Tabel 1. Sarnaste rakenduste loetelu

Rakendus	**Maksumus**	**Mõõtude sisestamine**	**Lõike konstrueeri-mine**	**Valmis lõige**	**Õpetus õmblemiseks**
Modaris (Lectra)	hinnapäring tuleb teha ettevõttele	Jah	Jah	Ei	Ei
TukaCAD	~18 eurot	Jah	Jah	Ei	Ei
Seamly 2D	tasuta	Jah	Jah	Ei	Ei
Burda	üks lõige 6 -12 eurot, raamatukogust võimalik tasuta laenutamiseks	Ei	Ei	Jah	Väga lühike
Käesolev töö	tasuta	Jah	Ei	Jah	Jah

Antud tabelist kajastub, et inimestel, kellel puudub eelnev lõigete konstrueerimise ja õmblemise kogemus, on loodav veebirakendus vajalik. Peamiseks põhjuseks on valmis lõigete olemasolu ja juhend edasisteks sammudeks. Loodava programmi suureks eeliseks on kindlasti ka aja kokkuhoid, kuna kasutajal pole vaja hakata ise lõiget konstrueerima ja saab asuda koheselt lõike välja printimisele ja õmblemisele.

4 Veebirakenduse analüüs

Analüüsi käigus määratletakse esmalt ära tulevase rakenduse funktsionaalsed ja mittefunktsionaalsed nõuded. Seejärel analüüsitakse erinevaid programmeerimiskeeli ja kliendi- ning teenusepoolseid rakendusi. Viimasena analüüsitakse erinevaid andmebaase. Analüüsi tulemusena valitakse kõik rakendused ja tehnoloogiad vastavalt tulevase rakenduse vajadustele.

4.1 Rakenduse nõuded

Rakenduse sihtgrupiks on kõik õmblemishuvilised inimesed, kes vajavad toetavaid juhendeid kogu protsessi vältel või kes soovivad saada lõike vastavalt enda mõõtudele. Veebirakendus on kindlasti suureks abimeheks ka kooli käsitöötundides, olukorras kus õpetajal puudub võimalus kontaktõpet läbi viia või õpetajal ajapuuduse tõttu ei ole võimalik piisavalt juhendada õpilasi. Käesolev rakendus peaks olema piisavalt selgete juhistega, et ka kooliõpilane suudaks neid järgida.

Loodavasse rakendusse tuleb esimesse versiooni kaks rolli: kasutaja ja administraator. Tulevikus on plaan jagada kasutajad kaheks, vastavalt sellele, kas kasutaja on liitunud tasulise versiooni või tasuta versiooniga. Tasuline versioon annab võimaluse kasutajal lisada peale enda kehamõõtude ka teiste inimeste mõõdud ja lasta programmil välja arvutada lõike suurused. Praeguses versioonis saab iga kasutaja lõike puhul lasta vaid korra programmil välja arvutada tulevase lõike suurused.

4.1.1 Funktsionaalsed nõuded

Tavakasutaja:

- Soovin registreeruda
- Soovin registreeruda läbi teiste rakenduste (gmail, facebook)
- Soovin sisse ja välja logida
- Soovin sisestada ja muuta enda kehamõõdud

- Soovin muuta enda parooli ja andmeid
- Soovin näha erinevaid juhendeid
- Soovin alla laadida lõikeid
- Soovin näha enda lõpuni läbitud õmmeldud tooteid
- Soovin näha enda alustatud tooteid
- Soovin valida detaili toote juurde
- Soovin vahetada rakenduse keelt

Administraator:

- Soovin sisse ja välja logida
- Soovin muuta enda andmeid
- Soovin lisada ja eemaldada administraatoreid
- Soovin hallata kasutaja õigusi
- Soovin lisada uusi lõikeid ja õpetusi
- Soovin kustutada lõikeid ja õpetusi
- Soovin teha muudatusi lõigete ja õpetuste juures
- Soovin ülevaadet veebirakenduse külastajate arvust
- Soovin vahetada rakenduse keelt

4.1.2 Mittefunktsionaalsed nõuded

Tavakasutaja:

- Soovin kasutada veebirakendust igas brauseris
- Soovin, et minu konto ja andmed oleksid turvalised

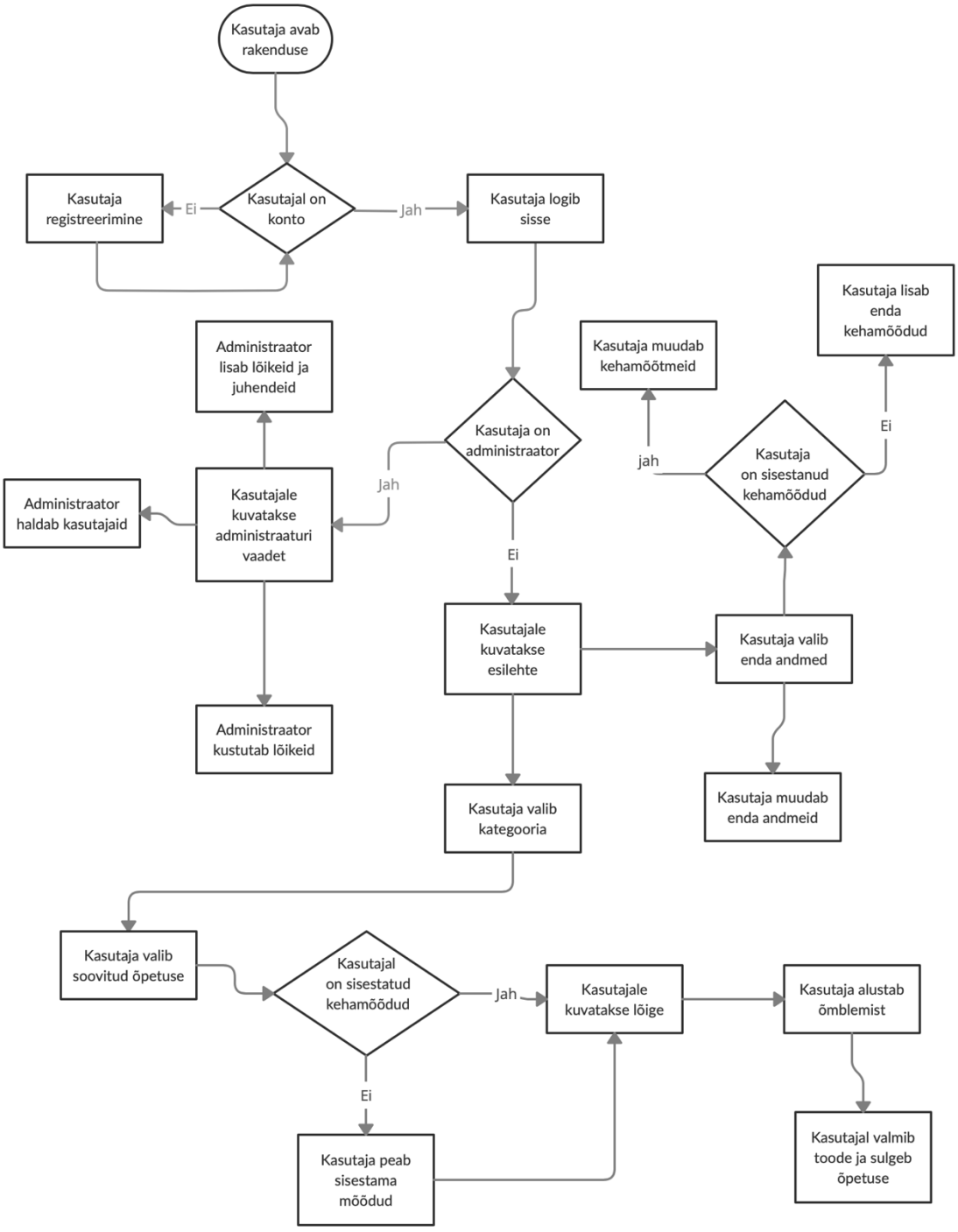
Administraator:

- Soovin kasutada veebirakendust igas brauseris

4.2 Kasutajalood

Kasutajalugude põhjal on lihtsam demonstreerida loodava rakenduse struktuuri. Kasutajalugudes on toodud välja kaks kasutajagruppi: tavakasutaja ja administraator. Kuna erinevatel kasutajagruppidel on erinevad võimalused, siis on toodud Joonis 2Joonis 2. Kasutajalugude skeem mõlema kasutaja võimalused. Kasutajalugudes on kirjeldatud

kogu protsessi alates sisselogimisest või registreerimisest ja seejärel juba erinevatest võimalustest, mis kasutajal avaneb sisse logitud vaates.



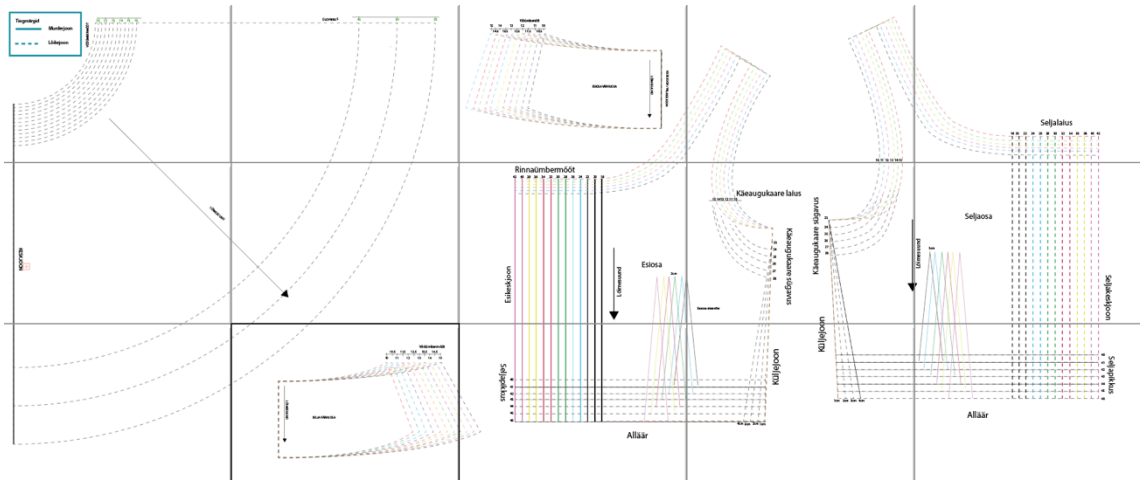
Joonis 2. Kasutajalugude skeem

Peamine erinevus kahe kasutajagrupi vahel on, et administraator saab lisada uusi lõikeid ja õpetusi ning neid hallata. Administraatoril on võimalus lõigete lisamisel lisada ka

mõõtudele avaruslisa. Avaruslisa on oluline lisada, kui etteantud lõike puhul ei ole see kehasse töödeldud. Seda näiteks riiete puhul, mis on mõnest kohast laiemad kui inimese enda keha. Administraator saab valida, millise mõõdu puhul ja kui palju avaruslisa tuleb jätta. Seejärel saab juba programm teha arvutused vastavalt sellele, kas avaruslisa on ette nähtud või ei. Administraatori ülesandeks jääb veel valida lõike juures kõik kasutatavad mõõdud, mida programmil on vaja arvutuste tegemisel. Lisaks tegeleb administraator kasutajate ja rollide haldamisega. Tavakasutajal puudub võimalus lõigete lisamiseks, kuna sellega soovitakse välistada vigaste lõigete ja õpetuste sattumist keskkonda. Erinevalt administraatorist on tavakasutajal võimalus sisestada enda keha mõõdud ja saada juhendit tegema asudes vastavad arvutused.

4.3 Lõiked

Loodava rakenduse üheks olulisemaks osaks on lõiked. Lõiked on lisatud rakendusse pdf kujul, mida on võimalik kasutajal alla laadida. Pdf-is on toodud lõige, mida kasutaja on valinud. Fail koosneb A4 lehtedest. Lõikefaili näidis on kujutatud Joonis 3. Lõikes on kujutatud erinevaid suurusi. Kasutaja ülesandeks saab vaid programmi poolt tagastatud suuruste leidmine lõikelt ja nende ühendamine joonega. Kasutajal on võimalus sama lõike isikupärastatud mõõtmed programmi poolt välja arvutada vaid korra, kuna sellega soovitakse välistada, et kasutaja hakkab õmblema teistele isikutele. Tulevikuplaanidesse kuulub soov lisada rakendusse tasulise kasutaja võimalus, mille puhul on võimalik kasutajal sama lõike puhul lisada erinevad kehamõõte, see võimaldab juba kasutada programmi ärilistel eesmärkidel. Kõikide lõigete konstrueerimisel on lähtutud raamatu “Rõivaste konstrueerimine ja õmblemine” [5] õpetustest.



Joonis 3. Lõike näidis

4.4 Rakenduse tuluallikas

Iga rakenduse ülalpidamise ja haldamisega on üksjagu kulutusi. Kui soovida, et veebirakendus oleks jätkusuutlik, tuleks mõelda võimalustele, et antud rakendus tooks sisse tulu. Käesolevas veebirakenduses on plaan luua kahte liiki kasutajaid ning lisaks administraator. Kasutajate erinevus seisneks rakenduse funktsionaalsuses. Esimese kasutajagrupi ligipääs materjalidele oleks tasuta, kuid kasutajal pole võimalus rakendada erinevaid kehamõõte sama lõike puhul mitu korda. Lisaks oleks kasutajal piiratud ligipääs mõne raskema õppematerjali juurde. Tasulisel kasutajal oleks olemas need eespool mainitud võimalused.

Samuti on planeeritud reklaamipindade müük veebirakenduses, mis oleks samuti hea võimalus teenida piisavalt tulu, et maksta rakenduse ülalpidamiskulude eest. Ka sponsorite otsimine on hea viis rahaliste probleemide lahendamiseks. Esialgselt oleks plaanis leida sponsorid sarnases kategoorias tegelevate valdkondade hulgast nagu näiteks kangapoed. Kuna antud rakendus oleks sobilik ka toetavaks abimaterjaliks koolide käsitöötundides, siis on võimalus otsida sponsoreid ka haridusasutuste hulgast.

4.5 Valitud tehnoloogiad

Tehnoloogiate valikute puhul on väga oluline arvestada tulevase rakenduse nõudeid ja suurust. Kuna programmeerimiskeelte ja raamistike hulk on mitmekesine, siis lõputöö

autor keskendub peamiselt tehnoloogiatele, millega on autor kokku puutunud, kas õpingute vältel või töötamise ajal.

Käesolevas lõputöös on äriloogika eraldatud kasutajaliidesest, tänu sellele on kood palju paindlikum. Eraldatud kasutajaliides ja äriloogika aitavad hiljem lisada lihtsamalt erinevaid liidestusi rakendusele. Kui on soov vahetada välja täielikult kasutajaliides, siis pole vaja uuesti kirjutada äriloogikat ja see hoiab kokku palju aega ja raha [6]. Ka arendamine ja testimine on tänu sellele mugavam.

4.5.1 Teenusepoolse programmeerimiskeele valik

Programmeerimiskeele valik on väga oluline ja kuna valikust sõltub ka raamistike valik, siis tuleks alustada arendamist just programmeerimiskeele valikuga. Peamised kriteeriumid teenusepoolse tehnoloogia valikus on kiirus, töökindlus ja mastaapsus.

Tuntumad teenusepoolsed tehnoloogiad veebirakenduste jaoks.

- Java - Java on üks populaarsemaid keeli, millel on laialdane kasutusala. Peamisteks eelisteks Java puhul on väga kõrge turvalisuse tase, kuna Javas on mitmeid sisseehitatud turvafunktsioone nagu näiteks autentimine ja juurdepääsukontroll. Lisaks on Java väga skaleeritav ja saab hakkama suurenenud töökoormusega [7]. Kuna Java on objektorienteeritud programmeerimiskeel, siis tänu sellele on rakendust lihtsam hallata ja see muudab kogu süsteemi paindlikumaks ja skaleeritavamaks [8].
- PHP - PHP on väga kiiresti omandatav ja seega sobilik ka algajatele. Lisaks on PHP-l olemas veel väga hea dokumentatsioon. Kuna PHP on vabavaraline, siis hoitakse arenduse kuludelt kokku. Suureks eeliseks on ka kiirus. Enamikel juhtudel võib PHP olla kuni kolm korda kiirem kui Python. Puudustena võib välja tuua huvi languse, mõningaste kaasaegsete teekide puudumise ja turvavead [9].
- JavaScript/NodeJS - JavaScript on kogunud väga kiirelt populaarsust ja muutunud üheks populaarsemaks programmeerimiskeeleks. JavaScriptis on võimalik teha valmis nii teenusepoolne kui ka kliendipoolne rakendus. Kuna NodeJS on väga kerge tehnoloogiaga, siis tänu sellele kiirendab see arenduse protsessi ja võib hoida kokku arenduse kuludelt [10].

- Python - Python on samuti maailmas laialdast kasutust leidnud. Lisaks on Pythonit lihtne õppida ja väga sobilik algajatele, kuna kood on lihtsasti loetav. Jõudluse ja kiiruse poolest jääb Python NodeJS-ile alla.
- C# - Objekt-orienteeritud programmeerimiskeel. C# on lihtne õppida, pole küll nii lihtne kui Python, aga siiski peetakse C# üsna lihtsasti õpitavaks. See võimaldab luua igas raskusastmes rakendusi. Peamine lihtsus seisneb korduvkasutatavas koodis. Miinusena võib välja tuua muudatuste korral koodi taaskäivitamise, mis võtab lisaaega [11].

Lõputöös on käsitletud vaid programme, millega lõputöö autoril on eelnev kogemus. Käsitletavad programmeerimiskeeled on kõik laialdaselt levinud ja kõigil on suur kogukond ning internetist leiab laialdaselt materjale. Kirjeldustest ei selgunud ühtegi piisavalt suurt põhjust, miks ei peaks mõnda valikut kaaluma. Kuna lõputöö aeg on piiratud, on mõistlik peamiselt kaaluda programmeerimiskeeli, mida oskab töö autor kõige paremini. Autoril on kõige enam kogemust C# ja Java programmeerimiskeeltega.

Java ja C# keelte süntaks on üsna sarnane. Mõlemate keelte puhul peetakse neid üsna lihtsasti õpitavaks. Mõlemal on mitmepalgeline arhitektuur, mis võimaldab käivitada mitut protsessi korraga. Veebi võrdlusnäitajate poolest kipub C# olema natuke parem, sest selle reageerimisaeg on lühem ja see võtab vähem CPU koormust [12]. Üldiselt nii C# kui Java on oma omaduste poolest sarnased. Java on veidikene populaarsem aga samal ajal on Java ka palju vanem kui C#. Mõlemal on väga lai kogukond ja samuti teekide hulk.

Lõpliku valiku tegemisel lähtus autor enda kogemustest. Kuna lõputöö autoril on suurem kogemus C# kasutamisel, siis just selle tõttu langes valik C# kasuks. Piiratud aja tõttu võib Java väiksem kogemus mängida olulist rolli ja antud lõputöö raames ei ole mõistlik hakata omandama uusi teadmisi, mis on juba omandatud C# kasutamisel.

4.5.2 Teenusepoolse raamistiku valik

Teenusepoolse raamistiku valik sõltub suuresti programmeerimiskeele valikust. Kuna eelpool mainiti, et programmeerimiskeeleks valiti C#, siis raamistiku puhul langeb koheselt valik .NET-i kasuks, kuna see on mõeldud just C# keele jaoks.

.NET on vabavaraline tarkvararaamistik, mille on välja töötanud Microsoft. Teisisõnu võib ka öelda, et .NET on virtuaalmasin erinevate programmide kompileerimiseks ja käivitamiseks [13]. Peamised keeled, mis kasutavad .NET-i raamistikku on C#, F# ja Visual Basic [14].

4.5.3 Kliendipoolse tehnoloogia valik

Kliendipoolse raamistiku eesmärk on kuvada kasutajale graafilist liidest kasutades selleks CSS-i, HTML-i ja Javascript-i.

Antud peatükis tuuakse välja populaarsemad kliendipoolsed raamistikud ja raamistikud, millega on autor ka ise kokku puutunud.

- React - React on avatud lähtekoodiga Javascripti raamistik loodud Facebooki poolt. React on aastaid olnud vaieldamatult üks populaarsemaid raamistikke kliendipoolsete rakenduste loomisel. React kasutab virtuaalset DOM-i, tänu millele muudetakse vaid HTML-is muudetud osa ja see kiirendab veebilehe laadimist [15].
- Angular - Angular on arendusplatvorm, mis on ehitatud TypeScript-iga. Tegemist on komponendipõhise raamistikuga skaleeritavate veebirakenduste loomiseks [16]. TypeScript on tugevasti tüübitud objekt-orienteeritud keel, mis kompileerub Javascriptiks. Teisisõnu on TypeScript on nii programmeerimiskeel kui ka tööriistade komplekt [17].
- Vue.js - Vue.js on JavaScripti raamistik kasutajaliideste ehitamiseks. Vue puhul on nii loogika, HTML kui ka CSS stiilid kõik ühes failis [18].
- Aurelia - Aurelia on avatud lähtekoodiga Javascripti moodulite kogum. Aurelia on jagatud mooduliteks, mis põhinevad funktsioonidele. Aurelia pole saavutanud nii suurt populaarsust kui eelpool mainitud raamistikud ja seega on ka Aurelia kogukond palju väiksem ja probleemide korral ei pruugi olla informatsioon nii lihtsasti leitav.

Väljatoodud tehnoloogiatest puudub lõputöö autoril kogemus Angulariga. Kuna lõputöö autoril on kõige suurem kogemus Reactiga ja samuti React on vaieldamatult mitmeid aastaid olnud tipus kliendipoolsete tehnoloogiate seas, siis tänu sellele valib autor

kliendipoolseks tehnoloogiaks Reacti. Reactil on samuti kõige suurem kogukond ja internetist leiab ilma igasuguste probleemideta vastuseid igasuguste probleemide korral.

4.6 Andmebaasi valik

Veebirakenduse oluline osa on andmebaas, kus hoiustada kasutaja ja rakenduse andmeid. Seega on ka andmebaasi valik väga oluline. Andmebaase on mitmeid erinevaid liike, kuid veebirakenduses kasutatakse peamiselt relatsioonilisi andmebaase. Samuti on lõputöö autoril kogemus vaid relatsiooniliste andmebaasidega.

Relatsioonilises andmebaasis salvestatakse andmed tabelisse. Tabelid koosnevad ridadest ja veergudest. Tabelid on omavahel seotud seostega.

Järgnevalt on välja toodud tuntumad relatsioonilised andmebaasid ja Tabel 2 toodud välja andmebaaside maksumus ja autori kogemus:

- MySQL - MySQL on kõige levinum avatud lähtekoodiga relatsiooniline andmebaas. MySQL-i loetakse lihtsasti õpitavaks. Ei sobi väga suure andmebaasi jaoks [19]. Saab kasutada tasuta, kui seda ei kasutata ärilistel eesmärkidel.
- PostgreSQL - Avatud lähtekoodiga objekt-relatsiooniline andmebaas, mis kasutab ja laiendab SQL keelt koos paljude funktsioonidega [20]. Sobib hästi rakendustele, mis võivad kasvada suureks ja milles on palju keerukaid päringuid. Võrreldes MySQL-iga eelis, kui rakendusel on vaja teha palju sisestamisi andmebaasi, lubab rohkem andmetüüpe, kui MySQL-is ja võrreldes MySQL-iga on parem indekseerimine [21].
- Oracle SQL - Oracle SQL on suletud lähtekoodiga tasuline andmebaas. Sobib väga suurte andmebaaside jaoks. Õpilastele on saadaval tasuta versioon, mida peetakese raskemini õpitavaks kui MySQL-i [19].
- Microsoft SQL Server - see on tasuline relatsiooniline andmebaas, väga turvaline ja ei luba töötamise ajal andmebaasifailidega manipuleerimist. Peetakse väga lihtsasti õpitavaks [22].

Tabel 2. Andmebaaside võrdlus

Andmebaas	**Maksumus**	**Kogemus**
MySQL	Tasuta	Hea
PostgreSQL	Tasuta	Hea
Oracle SQL	Tasuline	Hea
Microsoft SQL Server	Tasuline	Väga hea

Lõputöö autoril on kogemus kõigi eelpool mainitud andmebaasidega. Kõige enam kogemust omab autor Microsoft SQL Serveriga. Kuna tulevase rakenduse rahalised ressursid on piiratud, siis valik langeb vaid tasuta andmebaasidele. Tasuta andmebaasidest oli välja toodud MySQL ja PostgreSQL. Autoril on mõlema andmebaasiga kogemus üsna sarnane ja seega otsustamisel ei arvestatud kogemust. PostgreSQL sobib suuremate ja keerukamate andmebaaside puhul. Samuti peetakse PostgreSQL-i paremaks võrreldes MySQL-iga, kus on vaja palju sisestada andmebaasi. Tulevasel rakendusel saab olema väga palju uute lõigete ja õppejuhendite sisestamisi. Seega otsustas autor PostgreSQL-i kasuks.

5 Veebirakenduse arendus

Käesolevas peatükis käsitletakse teenuse- ja kliendipoolse rakenduse arendust ja testimist. Samuti leiab rakenduse peamised vaated ja juhendi rakenduse kasutamiseks. Täpsemalt arenduses kasutatavate tehnoloogiate kohta leiab alljärgnevatest peatükkidest.

5.1 Versioonihaldus

Oluline osa arenduse juures on versioonihaldus, kus hoiustada ja jagada enda koodi. Versioonihalduse kasutamise teeb mugavaks, kui on vaja jagada enda koodi või koos töötada mitme arendajaga. Versioonihalduses luuakse iga üleslaadimise korral uus versioon ja see võimaldab minna tagasi erinevate versioonide juurde. Samuti saab vaadata, milline arendaja milliseid muudatusi on teinud [23]. Versioonihaldustarkvaraks valiti Git, mis on tasuta kasutamiseks kõigile. Kuna lõputöö autoril puudub igasugune kokkupuude teiste versioonihaldustarkvaradega, siis langes kohe valik Git-i versioonihalduse kasuks. Samuti on Git ülipopulaarne ja erinevate probleemide korral leiab kiirelt internetist vastuse tänu laiale kasutajaskonnale.

Koodihoidla on hea vahend hoida enda koodi internetis, millele pääsevad ligi volitatud isikud, või hoida kogu kood avalikuna. Koodihoidlatest populaarsemad on GitLab ja GitHub. Mõlemad on oma olemuselt üsna sarnased. GitLab on avatud lähtekoodiga, GitHub mitte. Valiku langetasin GitLab-i kasuks, kuna autori on kõik eelnevad projektid hoiustatud GitLab keskkonnas.

5.2 Teenusepoolse rakenduse arendus

Teenusepoolne rakendus on laialdaselt tuntud *backend* nime all, mis vastutab kogu äriloogika, andmete vahendamise andmebaasiga ja kogu suhtluse ja andmevahetuse kliendipoolse rakenduse ehk frontendi vahel. Lisaks toimub teenusepoolses rakenduses suhtlus serveri vahel. Rakenduse teenusepoolseks tehnoloogiaks valiti .NET raamistik 6.0

versioon. Arenduskeskkonnaks valiti JetBrains Rider, kuna autoril on sellega kõige enam kogemust ja JetBrains Rideril on suurepärane tugi .NET rakenduste loomiseks.

5.2.1 Rakenduse arhitektuur

Rakenduses kasutatakse kihilist arhitektuuri. Kihiline arhitektuur on üks tuntumaid tarkvaraarhitektuuri mustreid, mille eesmärk on jagada kood erinevateks osadeks. Igal kihil rakenduses on konkreetne roll ja vastutus. Tänu kihilisele arhitektuurile on koodi lihtsam hallata ja testida [24].

Rakenduses alustati domeenimudelite loomisega. Domeenimudelis kirjeldatakse ära andmebaasi väljad, mille järgi hiljem rakendus genereerib andmebaasi vajalikud väljad. Domeenimudelis määratletakse ära ka välja maksimaalne pikkus. Seda on oluline teha, et mitte lasta andmebaasi salvestada liiga suuri objekte. Näidis domeenimudelist on kujutatud Joonis 4.

```
public class Category : DomainEntityId
{
    [MinLength(2)]
    [MaxLength(500)]
    public string Name { get; set; } = default!;

    public ICollection<Instruction>? Instructions { get; set; }
}
```

Joonis 4. Domeenimudel

Ennem andmebaasimudeli loomist tuleb luua migratsioonifail. Seda saab luua Joonis 5 toodud käsuga, mis tuleb sisestada terminali. Seejärel genereerib programm migratsiooni kausta koos vastavate failidega, milles on kirjeldatud andmebaasiskeeme. Hilisemalt uusi muudatusi domeenimudelis sisse viies tuleks uuesti lisada Joonis 5 toodud käsklus ja seejärel võrdleb programm vana mudelit uuega ja teeb kindlaks erinevused [26].

```
dotnet ef migrations add --project DAL.App.EF --startup-project WebApp --context AppDbContext Initial
```

Joonis 5. Migratsioonifailide loomine

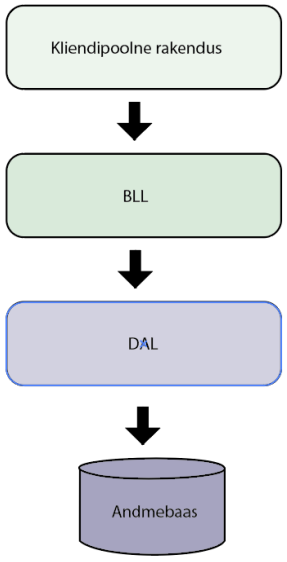
Andmebaasi muudatuste sisseviimiseks ja esmakordsel loomisel on vaja sisestada Joonis 6 käsklus terminali, mille järel programm loob või uuendab rakenduse andmebaasi. Seda

tuleks teha iga kord peale uue migratsiooni loomist. Seejärel on andmebaas loodud ja valmis kasutamiseks.

```
dotnet ef database update --project DAL.App.EF --startup-project WebApp --context AppDbContext
```

Joonis 6. Andmebaasi uuendamise käsklus

Andmete küsimiseks andmebaasist on võetud kasutusele hoidla muster (*Repository pattern*) ja UoW (*Unit of work)* muster. Hoidla mustrit kasutatakse, kui rakenduses on vaja ligi pääseda andmetele. Selle peamiseks ülesandeks on andmete salvestamine ja hankimine. Samuti hoidla mustri kasutamine vähendab koodi korduvkasutamist, kuna rakendusega seotud kõigi andmete pärimine on viidud ühte kohta ja pole vaja eraldi mujal samu päringuid teostada. UoW ülesandeks on teha andmebaasi lisamisel, muutmisel, pärimisel või kustutamisel vaid üks päring mitme päringu asemel. Kõik need toimingud kas läbivad või ebaõnnestuvad ühe üksusena [25]. Lisaks on loodud rakendusse andmete ligipääsemiseks DAL (D*ata Access Layer*) kiht. DAL kiht koondab endasse kogu rakenduse andmebaasi käskude väljastamise. Seega DAL kihis peaksid olema kõik andmebaasi sisestamise, kustutamise, muutmise ja pärimise käskudega seotud toimingud [26]. Rakendusele on lisatud veel BLL (*Business Logic Layer*) kiht. See kiht vastutab kõik äriloogikaga seonduvaga. Samuti on kõik koondatud ühte kohta, mis aitab vältida koodi kordumist. BLL kiht pärib DAL kihist kõik vajalikud andmebaasi andmed. Lihtsustav joonis rakenduse arhitektuurist on kujutatud Joonis 7.



Joonis 7. Rakenduse arhitektuur

5.2.2 REST API

Kliendipoolne rakendus suhtleb teenusepoolse rakendusega kontrollerite vahendusel. REST API kasutab andmete juurdepääsuks HTTP-päringuid. HTTP päringuga pöördutakse kontrolleri poole ja vastav päring pärib andmed äriloogika kihist. Peamised HTTP päringud on GET (päritakse andmeid), PUT (muudetakse andmeid), POST(lisatakse uued andmed) ja DELETE (kustutatakse andmed) [27]. .NET-il on väga mugav võimalus lasta programmil genereerida valmis kontrollerid, kus on kõik peamised meetodid olemas. Joonis 8 on kujutatud vastav käsklus, mis tuleb sisestada terminal. Hiljem saab juba enda vajadustele muuta valmis genereerituid meetoteid või lisada juurde uusi.

```
dotnet aspnet-codegenerator controller -name UnitsController -actions -m Domain.App.Unit -dc
AppDbContext -outDir ApiControllers -api --useAsyncActions  -f
```

Joonis 8. Api kontrolleri loomine

Joonis 9 on kuvatõmmis GET meetodist, mille abil tehakse päring kõigi väljade pärimiseks etteantud andmebaasi tabelist. Oluline osa on määrata ära, et tegu on GET päringuga.

```
// GET: api/BodyMeasurements
[HttpGet]
[Produces( contentType: "application/json")]
[Consumes( contentType: "application/json")]
[AllowAnonymous]
[ProducesResponseType(typeof(PublicApi.DTO.v1.BodyMeasurements), statusCode: StatusCodes.Status200OK)]
1 usage   hlepik +1
public async Task<ActionResult<IEnumerable<PublicApi.DTO.v1.BodyMeasurements>>> GetBodyMeasurements()
{
    return Ok((await _bll.BodyMeasurements.GetAllAsync()).Select(a :BodyMeasurements => _mapper.Map(a)));
}
```

Joonis 9. Näide Api kontrolleri get päringust

Kontrollerisse lisatakse iga meetodi juurde andmete formaat, mida võetakse vastu või millist tüüpi tagastatakse. Antud lõputöös on kasutusel kahte liiki andmete formaati. Nendeks on Application/json ja Multipart/form-data.

Application/json võtab vastu või tagastab andmed JSON-i kujul. JSON on JavaScriptil põhinev andmevahetusvorming, mida on lihtne lugeda ja mida saab kasutada iga

programmeerimiskeelega. JSON-il on ka sisse ehitatud funktsionaalsus teisendamaks andmed objektideks ja vastupidi [28].

Multipart/form-data-t kasutatakse lõputöös failide ja piltide saatmiseks. FormData-ga saadetakse andmeid, mis ei ole näiteks JavaSripti objektid. Enne saatmist muudetakse andmete võti - väärtus paarideks [29]. Hiljem salvestatakse kontrolleris failid ja pildid lokaalselt kaustadesse. Failide ja piltide hoiustamine lokaalselt on väga odav ja saab hoiustada suurt hulka. Faile ja pilte on võimalik hoiustada ka andmebaasis või pilves. Andmebaasis pole mõistlik hoiustada suurt hulka faile, kuna see viib andmebaasi kiiruse alla ja päringute tegemine võib hakata võtma tunduvalt rohkem aega. Pilves hoiustamine on enamjaolt tasuline, seega antud lõputöö raames leiti parimaks lahenduseks hoiustada pildid lokaalselt.

Tulevasele rakendusele on võimaldatud ka erinevate versioonide lubamine. Erinevate versioonide lubamine on hea viis muuta olemasolevat veebirakendust või lisada juurde uut funktsionaalsust veebirakendust lõhkumata. Erinevate versioonide lubamist rakenduses on mitu võimalust. Antud lõputöös on valitud URL kaudu edastatavad versioonid. See tähendab, et URL-i lisatakse päringu ajal hetkel kasutuses olev versioon [28]. Versiooni number on lisatud nii kontrollerisse kui ka kliendipoolse rakenduse päringute tegemisel. Kasutajale ei kuvata olemasolevat versiooni ja seega pole kasutajal aimdust hetkel kasutatavast versioonist.

5.2.3 Autentimine

Lõputöö veebirakenduses on võetud kasutusele ASP.NETi poolt loodud süsteem autentimiseks ja autoriseerimiseks kasutajaid. Tänu sellele on muudetud lihtsaks sisse - ja väljalogimine. Lisaks veel registreerimise ja erinevate rollide lisamine ja haldamine. Autentimisel kasutatakse info edastamiseks JWT (JSON Web Token). JWT võimaldab saata teabe osapoolte vahel turvaliselt JSON objektina. Saadetud info on usaldusväärne, kuna see on digitaalselt allkirjastatud. JWT koosneb kolmest osast, mis kõik on eraldatud punktiga. Esimene osa header ehk päis, mis koosneb kahest osast: märgi tüübist ehk JWT ja kasutatavast allkirjastamise algoritmist. Teine osa sisaldab infot kasutaja kohta. See võib sisaldada näiteks kasutaja nime ja rolli. Teises osas võib olla ka JWT aegumisaeg, mille järgselt pole võimalik enam seda kasutada. Viimaseks osaks on allkiri, mida kasutatakse selleks, et kontrollida kas sõnumit on saatmise ajal muudetud [32].

Käesolevas veebirakenduses on kasutusel erinevad rollid. Sõltuvalt rollile on veebirakenduse funktsionaalsus erinev. Kontrollerites saab ära määrata, kellel on õigust antud andmetele ligi pääseda või teostada muudatusi ja kustutamisi. Selleks lisatakse Joonis 10 toodud koodijupp kontrolleri meetodi päisesse. Toodud näitel määratakse ära, et vastava kontrolleri andmete pärimisele või muutmisele saab ligi vaid kasutaja rolliga "Admin". Kontrolleris saab lisada meetodite päisesse tingimuse, kellel on õigus sealt infot pärida või saata.

```
[Authorize(AuthenticationSchemes = JwtBearerDefaults.AuthenticationScheme, Roles = "Admin")]
```

Joonis 10. Kontrolleri juurdepääsupiirangu tingimus

5.2.4 Serverisse paigaldamine

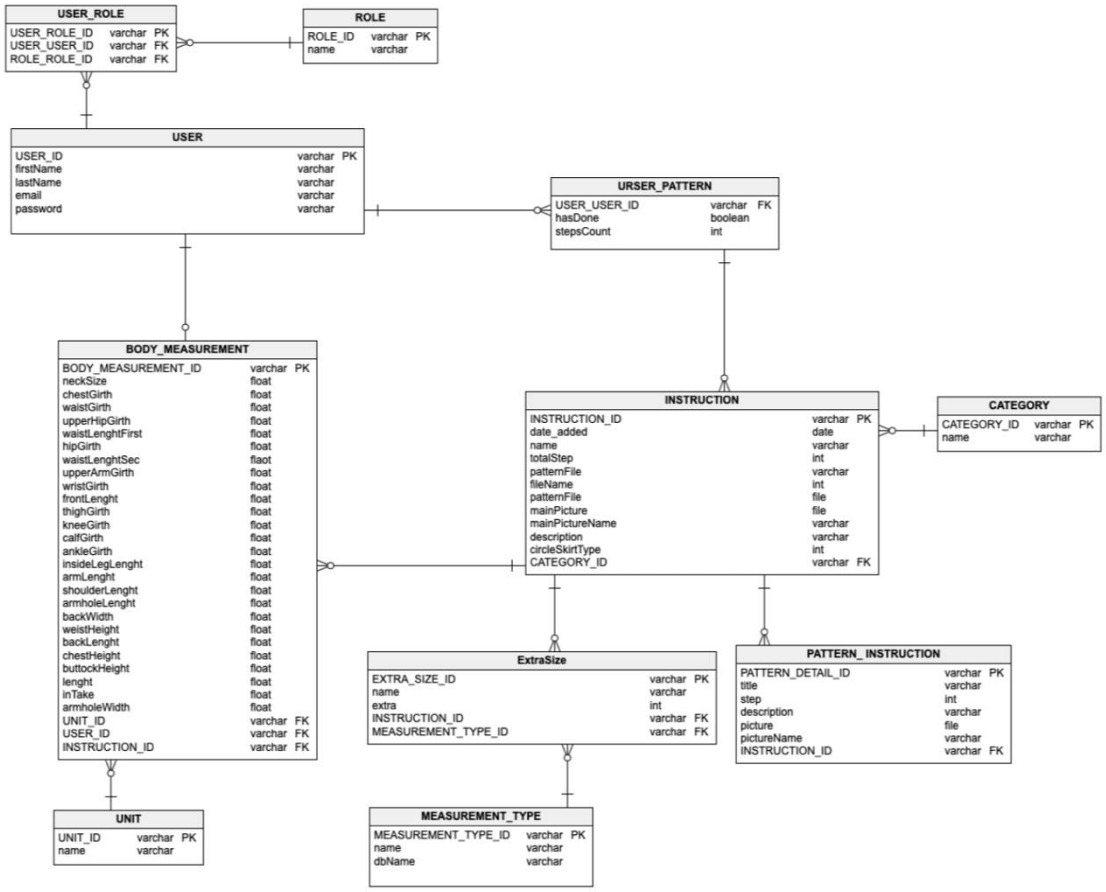
Käesolev rakendus paigaldati serverisse, et muuta see ligipääsetavaks ka teistele kasutajatele. Kõigepealt juba rakenduse arenduse alguses loodi Docker fail, milles määratleti kõik vajalikud käsud Dockeri pildi loomiseks. Docker on avatud lähtekoodiga tarkvaraplatvorm, mis aitab kiiresti ehitada ja testida rakendust. Docker pakendab rakenduse konteineriteks, mis sisaldab kõike vajalikku, sealhulgas teeke. Lisaks rakendusele paigaldati Dockerisse ka andmebaas. See võimaldas kasutajaliidese arendamise ajal kasutada juba Dockeris olemasolevat konteinerit ja polnud vaja enam käivitada teenusepoolset rakendust kasutajaliidese arendamise ajal. Docker failile lisaks on vaja luua docker-compose.yml. Docker-compose on tööriist defineerimiseks ja käivitamiseks mitme konteineriga Dockeri rakendusi. Tänu sellele on võimalik käivitada korraga kõik rakendused, mis on defineeritud docker-compose failis ühe käsuga [30].

Serverisse paigaldamiseks tuli luua virtuaalmasin ja seejärel sinna lisada Docker. Kuna juba arenduse alguses loodi vajalikud docker failid, siis virtuaalmasinas polnud vaja enam muud teha kui luua Dockerisse vajaminevad konteinerid. Kui konteinerid said loodud, siis oligi rakendus valmis juba serveris kasutamiseks ja kättesaadav kõigile.

5.2.5 Andmebaas

Andmebaas on väga oluline osa enamike veebirakenduste puhul. Andmebaasi puhul tuleks esmalt ära määratleda loodavad tabelid ja veerud. Lihtsamaks ettekujutluseks loodavast andmebaasist on mõistlik see esmaselt üles joonistada. Lõputöö autor on kasutanud Vertabelo veebirakendust kujutamaks tulevast loodavat andmebaasi. Järgneval

Joonis 11 on kujutatud olemi-suhte diagrammi. Lisatud on nii andmebaasi tabeli kui veergude nimed ja kujutatud tabelite omavahelisi suhteid.



Joonis 11. Andmebaasi mudel

.NET-is rakenduste jaoks on loodud Entity Framework raamistik, mis võimaldab arendajatel töötada andmetega kasutades domeenispetsiifiliste klasside objekte [31]. EF kasutab omakorda suhtlemiseks LINQ päringuid. LINQ on tehnoloogiate kogum, mis võimaldab päringud tõlgendada otse C# keelde. Tänu LINQ-ile on pärimine andmebaasist tehtud kasutaja jaoks väga loetavaks ja minimaalse koodiga saab teostada näiteks filtreerimisi ja otsimisi [32].

5.3 Kliendipoolse rakenduse arendus

Kliendipoolse rakenduse arendus on eraldiseisev teenusepoolsest rakendusest ja loodud React raamistikku kasutades, mille kasuks langes valik analüüsi osas. Arenduskeskkonnaks otsustati valida JetBrains-i poolt pakutav WebStorm. Otsuse langetamisel peamiseks põhjuseks sai juba tuttav keskkond, kuna teenusepoolne rakendus

loodi samuti JetBrains-i poolt pakutavas keskkonnas, siis WebStormi programmi ülesehitus on pea üks-ühele ja seega nii teenusepoolse kui kliendipoolse rakenduse arendus on mugavam, kui arenduskeskkond on samasugune.

5.3.1 Üheleherakendus

Üldiselt tuntakse kahte võimalust luua kliendipoolne rakendus. Nendeks on SPA (*Single Page Application*) ehk üheleherakendus või MPA (*Multi-page Application*) ehk mitmeleherakendus. Erinevus seisneb selles, et SPA puhul töötab rakendus brauseris ja ei nõua lehe uuesti allalaadimist kasutamise ajal. Lehel muudatuste korral laetakse vaid muutunud osa ja ei laeta kogu lehte uuesti. Mitmeleherakenduse puhul laetakse muudatuste korral leht uuesti alla. Käesolevas lõputöös otsustati kasutada SPA lahendust, kuna SPA reageerimisaeg on kiirem kui MPA-l. Samuti peetakse üheleherakenduse arendamist lihtsamaks [33].

5.3.2 Komponendid

Arenduse käigus loodi ka mõningad komponendid. Komponendid on parim viis vähendada koodi kordust ja lisaks on see hea viis kujundamaks rakendust. Sõltuvalt vajadusest saab komponentidele anda kaasa parameetreid, kuid saab ka ilma. Näiteks kui soovitakse kuvada samasugust tekstikasti erinevatel lehtedel, aga erineva sisuga, siis on võimalik parameetrina anda kaasa teksti sisu ja pole vaja luua igale lehele sama tekstikasti, vaid saab kasutada valmistehtud komponenti. Komponentidega on ka hea võimalus muuta kujunduselemente. Näiteks saab luua nupu komponenti ja anda kaasa nupu teksti ja kujunduselemente, sealhulgas värvus, suurus, ikoon. Komponentide kasutamine lihtsustab rakenduses kirjutatavat koodi. Joonis 12 on kasutatud sama komponenti andes kaasa erinevad parameetrid. Antud joonis annab suurepärase ülevaate, kuidas on kasutatud sama komponenti ja saadud erinev tulemus.

Joonis 12. Komponendi kasutamine nuppude puhul

React pakub lisaks ka hulgaliselt teeke, mis hõlbustavad veebilehe disainimist ja arendamist. Soovitud teegid tuleb enne kasutamist alla laadida ja seejärel leiab nende seast juba hulgaliselt valmisehitatud komponente ning arendajal pole vaja siis ise kõike

nullist luua. Üks enim kasutatud Reacti kasutajaliidese teeke on Material UI. Samuti otsustas töö autor antud lõputöös just Material UI kasutada, tänu selle suurele populaarsusele ja varem omandatud kogemusele antud teegi kasutamisel.

5.3.3 Suhtlemine serveripoolse rakendusega

Kliendipoolne rakendus saab serveripoolsest rakendusest info kätte HTTP päringute abil. Päringute tegemiseks on Reactis vajalik installida Axios teek. Axios teek teeb lihtsaks HTTP päringute saatmise REST lõpp-punktidesse ja CRUD toimingute sooritamise [38]. Päringuga koos lisatakse päisesse kasutaja info ja saadetava sisu tüüp. Lisaks tuleb lisada API endpoint, mis määrab asukoha, kuhu päring saadetakse [39]. Teisisõnu API endpoint määrab ära millise serveripoolse rakenduse kontrolleri meetodi ligi me soovime pääseda. Kuna päringu päisesse lisatakse alati JWT, siis tänu sellele saab kontrolleris rakendus veenduda, kas antud isik on volitatud tegema vastavat päringut või ei. Kui kasutajal puudub volitus, siis tagastab serveripoolne rakendus veateate ja seejärel saab kasutajale kuvada veateate, kui suunata logimise lehele.

5.3.4 Turvalisus

Iga rakenduse oluline osa on turvalisus, et välistada andmete leket ja rakenduse toimimist. Käesolevas lõputöös saab sisestada tekstiredaktorisse teksti, mis on HTML kujul. Hiljem selle kuvamine võib põhjustada turvariske. Selle vältimiseks on appi võetud teek dompurify, mida tuleb ennem kasutamist installida. Dompurify puhastab HTML-i eemaldades ära HTML märgendid ja lisaks aitab hoiduda XSS rünnakute eest. XSS (*Cross-site scripting*) rünnak on üks levinumaid, mille käigus pahatahtlik kood süstitakse veebilehele ja seejärel käivitatakse. Kasutaja sisestab tekstivälja vastava koodi ja kui rakendus seda lehel kuvab, siis kasutaja sisestatud kood käivitub [40]. Seega on väga oluline, et kui vaja on HTML-i kuvada, siis ei tohiks kindlasti seda otse teha ja tuleks kasutada näiteks dompurify teeki, mis eemaldab XSS rünnakuga seotud riskid.

Teenusepoolse rakenduse arenduse juures räägiti REST API kasutamisest. REST API loob täiendava turvakihi teenusepoolse ja kliendipoolse rakenduse vahele ja tänu sellele aitab see välistada SQL süstimise rünnaku eest. SQL süstimise rünnaku puhul, lisatakse andmebassi käsklus andmete salvestamise ajal [41]. Andmebaasis see käsklus käivitub ja tänu sellele on võimalik näiteks kustutada, pärida ja muuta andmebaasis olevaid andmeid.

Ründaja võib sellega seoses tekitada väga suurt kahju ja kasutajate andmed võivad lekkida. Seega on väga oluline, et kasutaja ei saaks otse lisada andmeid andmebaasi.

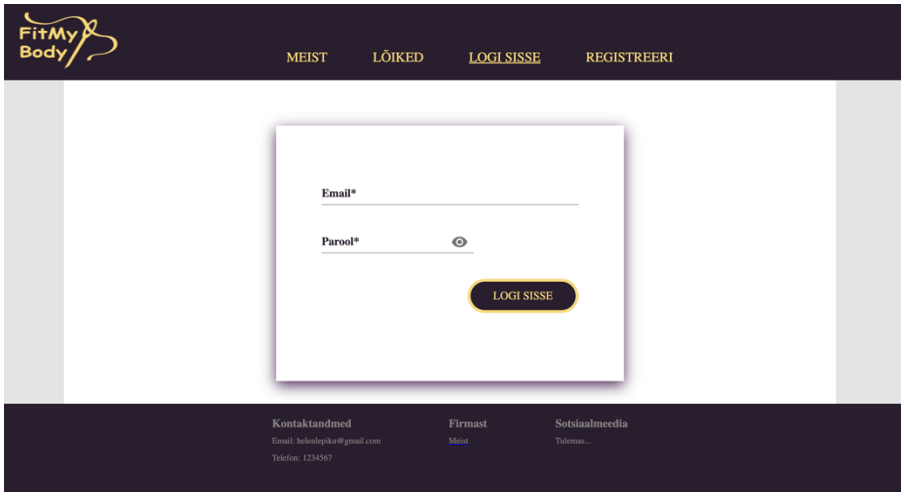
Käesolevas rakenduses on vaja sisestada ka hulgaliselt pilte ja faile. Väga oluline on kontrollida ja piirata, et sisestatud failid vastavad nõuetele. Samuti tuleks piirata failide suurust, et kasutaja ei saaks sisestada liiga suuri faile ja sellega koormata üle kogu andmebaasi. Seega on käesolevasse rakendusse lisatud kontroll, et sisestatud failitüübid oleksid vaid need, mis on ette määratud ja nende maht ei ületaks lubatut. Kasutajale kuvatakse koheselt veateade, kui failid ei vasta nõuetele.

5.4 Rakenduse disain

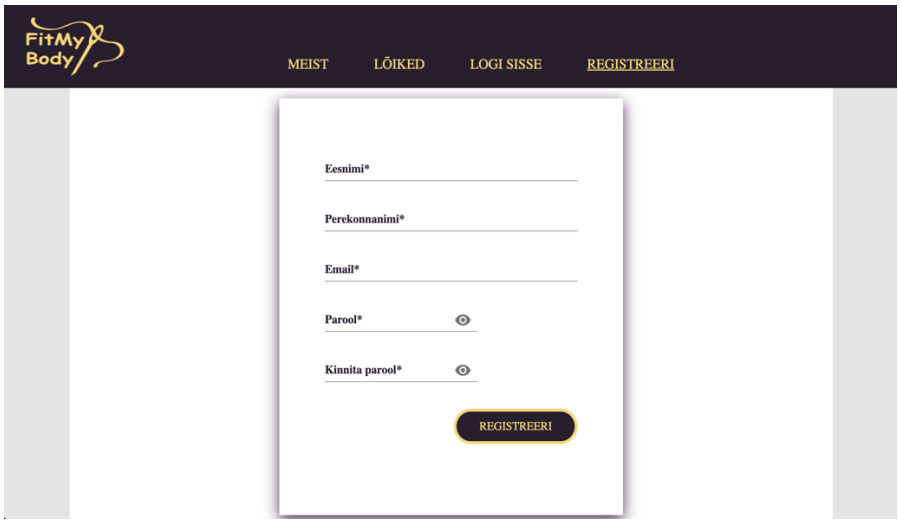
Käesolevas peatükis leiab rakenduse erinevaid vaateid ja juhendid rakenduse kasutamiseks. Lisatud on administraatori kui tavakasutaja vaateid.

5.4.1 Sisselogimine ja registreerimine

Kasutajad saavad tutvuda rakendusega ilma kontot omamata, kuid kui kasutaja soovib endale alla laadida lõike ja järgida õpetust, siis selleks peab kasutaja olema registreeritud ja sisse logitud, et programm saaks tuvastada kasutaja. Joonis 13 ja Joonis 14 näeb nii sisselogimise kui registreerimise vaadet.



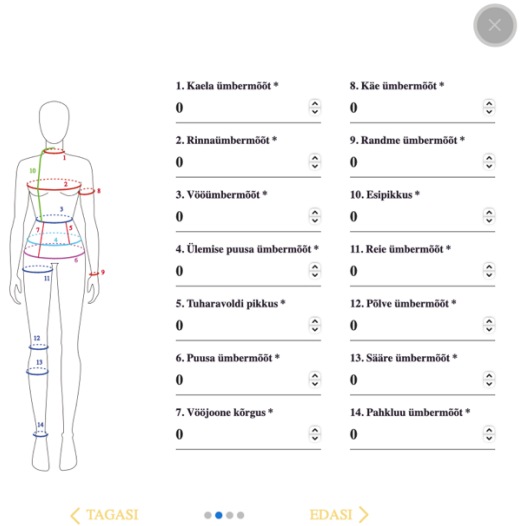
Joonis 13. Rakenduse sisselogimise vaade



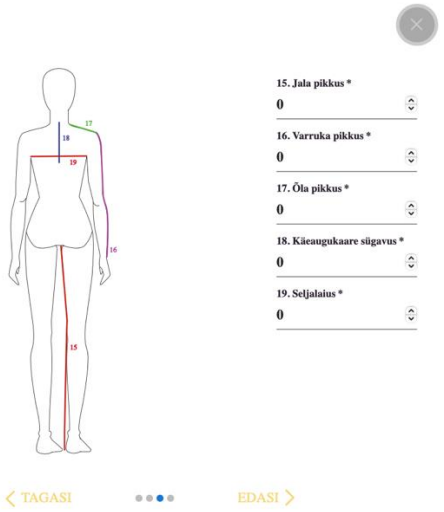
Joonis 14. Rakenduse registreerimise vaade

5.4.2 Kasutaja kehamõõtude sisestamine

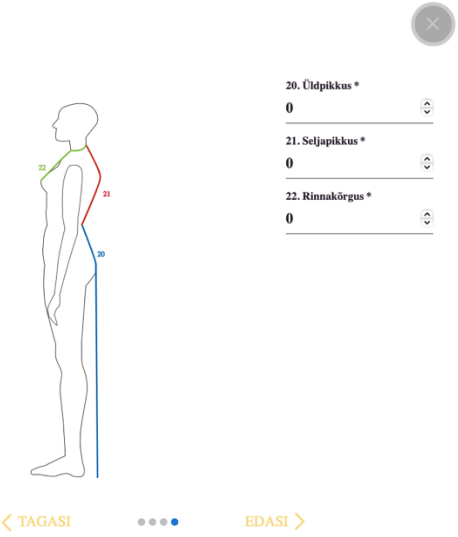
Kehamõõtude sisestamisel leiab kasutaja abistavaid pilte, mille järgi on lihtsam vastavad mõõdud endalt võtta. Kehamõõtude sisestamisel peab kasutaja valima ka mõõtühikud. See on oluline just eri rahvuste jaoks, et muuta neile kasutamine mugavaks, et nad saaksid kasutada mõõtühikuid, millega nad on harjunud. Lisaks kehamõõtude sisestamisel on lisatud kontroll, et kõik mõõdud oleksid sisestatud. See teeb hilisemalt arvutamise lihtsamaks ja pole vaja enam kontrollida, kas kõik mõõdud on olemas. Kasutajale on see samuti mugavam, vaid korra sisestab ta enda mõõdud ja ei pea enam tulema tagasi, kui mõni mõõt on puudu. Alljärgnevatel piltidel Joonis 15, Joonis 16, Joonis 17 ja Joonis 18 on toodud kuvatõmmised rakenduse mõõtude sisestamise vaadetest.



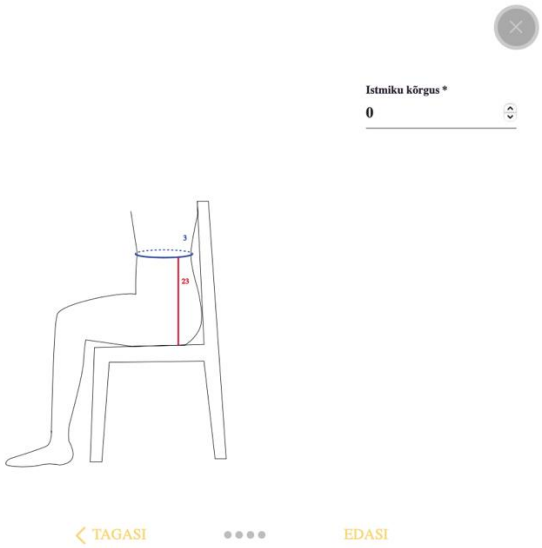
Joonis 15. Kasutaja kehamõõtude sisestamine (esivaade)



Joonis 16. Kasutaja kehamõõtude sisestamine (tagavaade)



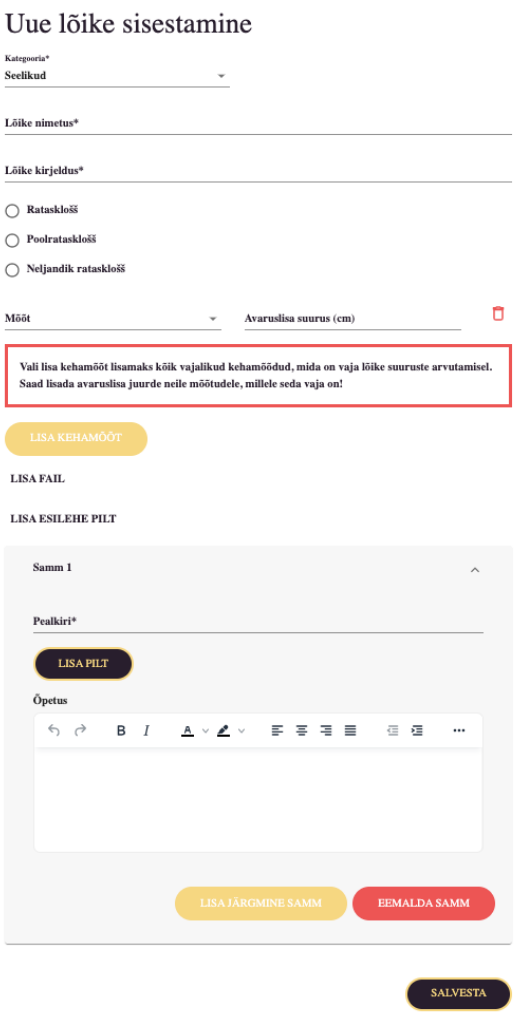
Joonis 17. Kasutaja kehamõõtude sisestamine (külgvaade)



Joonis 18. Kasutaja kehamõõtude sisestamine (istumise vaade)

5.4.3 Lõike sisestamise vaade

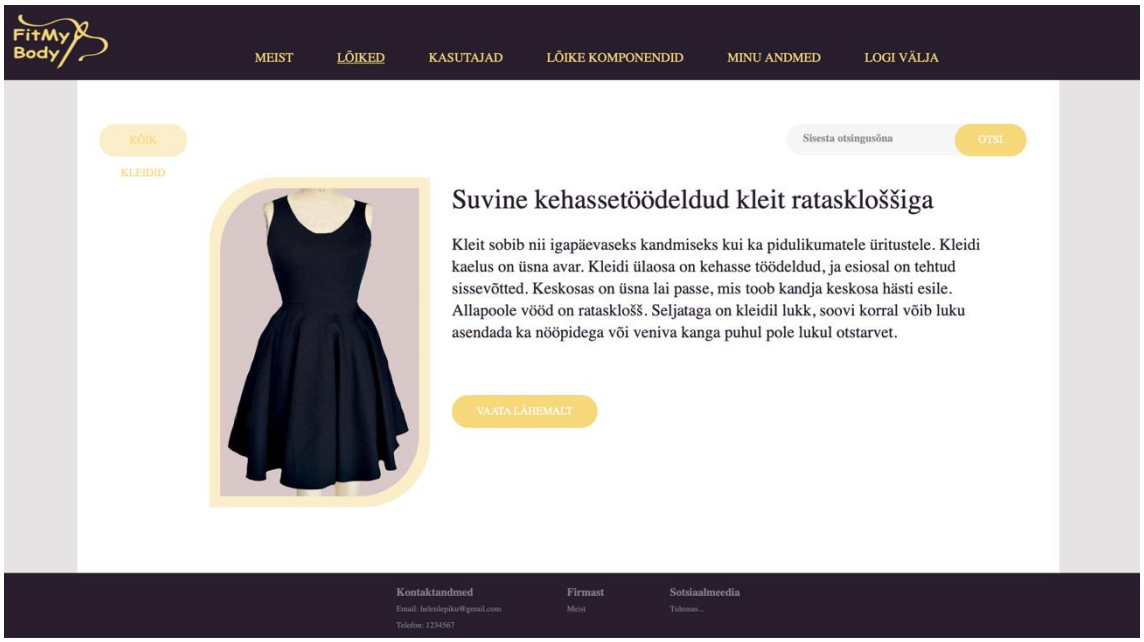
Lõike sisestamine ja juhendi lisamine on pikk protsess ja on üritatud teha võimalikult mugavaks. Näide lõike sisestamisest on leitav Joonis 19. Esmalt saab administraator lisada lõike nimetuse ja kirjeldava teksti, seejärel on kohustuslik lisada nii esipilt kui ka lõikefail. Esipilti kuvatakse alati lõike kirjelduse juures ja see annab kasutajale aimduse, millise tootega on tegu. Lõikefaili puhul on lubatud vaid pdf formaat. Lisaks tuleb administraatoril lisada vajaminevad mõõdud, et hilisemalt programm teaks, milliseid mõõte arvutamisel vaja on. Administraator saab lisada iga mõõdu juurde avaruslisa, see on oluline riiete puhul, mis ei ole inimese kehasse töödeldud ja on laiemad. Kui valitakse kategooria seelikud või kleidid, siis avaneb täiendav valikuvõimalus. Valida saab kolme valiku vahel, et kas lõikes on kasutatud ratasklošši, poolratasklošši või neljandik ratasklošši. Valik pole kohustuslik ja kui lõikes pole kasutatud eelpool mainitud valikuid, siis programm arvutab vastavalt puusaümbermõõdu ja vööümbermõõdu järgi. Kui aga on kasutatud nimetatuid valikuid, siis arvutamine toimub teisel viisil ja programmile on vaja ette öelda, millise valemi järgi tuleb vastavad arvutused teha. Viimasena saab administraator lisada juhendi, vajutades lisa *uus samm* nuppu. Iga sammu juures saab administraator määrata pealkirja, lisada pildi ja teksti. Teksti puhul on lisatud tekstiredaktor, et vajadusel lisada erinevaid värve või stiile, tuues esile olulisemat teksti. Samuti on võimalik lisada näiteks tabeleid. Juhendi sammude hulk ei ole määratud ja administraatoril on võimalik lisada niipalju samme, kui heaks arvab.



Joonis 19. Uue lõike sisestamine

5.4.4 Otsingu vaade

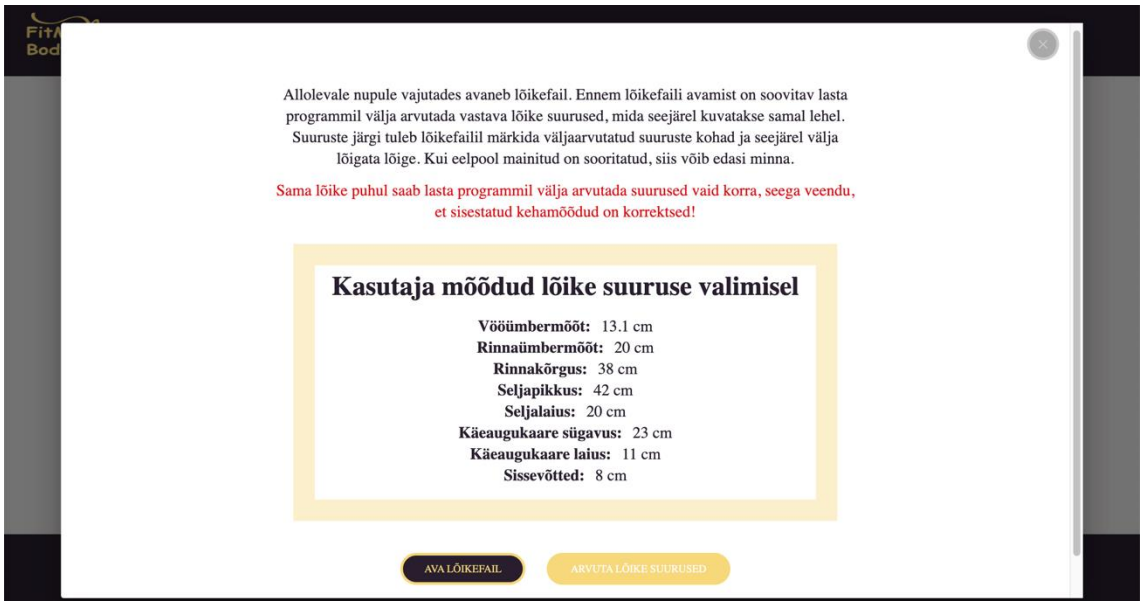
Otsingu vaates Joonis 20 kuvatakse kasutajale erinevaid kategooriaid. Kasutaja saab vastavalt kategooriale vaadata lõikeid ja otsinguriba abil otsida kindlaid märksõnu. Lisaks on kasutajal võimalus valida kõik kategooriad korraga, et lihtsustada märksõnade otsingut. Sobiva lõike leidmisel saab kasutaja selle avada.



Joonis 20. Lõigete otsimise vaade

5.4.5 Juhendi avamine

Juhendi avamisel Joonis 21 avaneb kasutajal esimese sammuna võimalus alla laadida lõikefail ja arvutada välja vastavad kehamõõdud, mida on vaja märkida lõikefaili juurde. Kui kasutaja on juba korra lasknud enda mõõdud välja arvutada, siis seda kordama enam ei pea, siis kuvatakse juba arvutatud mõõdud. Seejärel võib kasutaja liikuda edasi juhendi juurde. Kui kasutaja sulgeb juhendi ja pole sellega lõpuni jõudnud, salvestatakse pooleliolev etapp ja kuvatakse hiljem kasutaja lõigete loetelu tabelis. Joonis 22 on toodud kasutaja lõigete tabel, milles kuvatakse nii sooritatud kui ka pooleliolevad lõiked. See lihtsustab kasutajal kiiresti suunduda tagasi pooleliolevate lõigete juurde ja pole vaja enam otsinguriba kasutada, et leida üles vastav lõige.



Joonis 21. Juhendi alustamise vaade

Joonis 22. Kasutaja sooritatud ja mittesooritatud juhendid

5.5 Testimine

Üldiselt kuulub iga rakenduse juurde ka testimine. Testimise käigus võivad välja tulla vead, mida on eelnevalt arendajal jäänud märkamata, samuti muudatuste korral on testimine hea viis kontrollimaks, kas lahendus töötab endistviisi. Rakenduse testimiseks on loodud automaattestid. Testimisel kasutatakse nii ühiktestimist kui ka integratsiooni testimist.

Ühiktestimise puhul testitakse väiksemat koodijuppi, mida saab eraldada loogiliselt koodist. See võib olla näiteks mõni funktsioon, meetod või alamprogramm [34]. Integratsioonitestide puhul testitakse erinevaid osasid koos ja tihti kasutatakse testimisel ka andmebaasi. Lõputöö raames on loodud integratsiooni testimise jaoks mälupõhine andmebaas, et mitte muuta rakendusse juba sisestatud andmeid.

6 Valminud rakenduse analüüs

Valminud rakenduses sai valmis tehtud põhiline eesmärk, mida loodeti saavutada. Rakenduses loodi võimalus administraatoril sisestada uusi lõikeid ja õpetusi, võimalus muuta ja kustutada olemasolevaid lõikeid. Administraator saab hallata kasutajaid ja rolle, lisada uusi kategooriaid ja mõõtühikuid. Registreeritud kasutajatele anti võimalus vaadata juba läbitud või alles pooleliolevaid juhendeid. Lisaks on võimalus kasutajal sisestada enda kehamõõdud, mida programm arvestab uue lõike avamisel ja arvutab vastavalt sellele välja valitud lõike suurused. Kasutajale loodi võimalus ka muuta enda kehamõõte, kuid juba sooritatud lõike puhul pole kasutajal enam võimalust lasta programmil uuesti arvutada, sest antud funktsionaalsus on planeeritud tasulise kasutajana. Valminud rakenduse link on leitav lisa 1 alt.

Piiratud aja tõttu jäi tegemata mitmed funktsionaalsed nõuded, mis olid algselt planeeritud loodavasse rakendusse. Hetkel puudub võimalus kasutajal läbi teise rakenduse end registreerida ja sisse logida, mis oli algselt plaanitud. Samuti õmblemisprotsessi ajal puudub kasutajal valik, näiteks erinevate taskudisainide osas, mida loodeti samuti saavutada. Valminud rakendusse lisati kaks näidist kasutajale proovimiseks ja õpetuse läbimiseks. Algselt oli plaanitud lisada rohkem valikuid kasutajale, kuid kuna õpetuste lisamine ja lõigete valmistamine on väga ajamahukas protsess, siis piirtuti hetkel vaid paari näidisega, et anda edasi loodava rakenduse funktsionaalsust ja saada esmane tagasiside kasutajatelt.

6.1 Kasutaja tagasiside

Käesoleva rakenduse põhjaliku tagasiside saamiseks läheb üsna pikalt aega, kuna kogu protsessi läbimine võib aega võtta, sõltuvalt inimesest, mitmest päevast kuni nädalateni. Seega otsustati rakendust anda testimiseks inimesele, kes tegutseb antud valdkonnas. Testkasutaja õmbleb hobikorras ja annab käsitöö tunde koolis.

6.1.1 Mõõtude sisestamine

Mõõtude sisestamine oli testkasutaja meelest väga arusaadav. Joonised olid väga lihtsad ja selged. Samuti numbrid aitasid aru saada, mis kohast tuli järgnev mõõt võtta. Kasutajale meeldis, et mõõtude võtmine oli mitmele lehele ära jaotatud ja ühel lehel polnud liiga palju eri mõõte koos ja see muutis arusaamise lihtsamaks.

6.1.2 Lõike välja lõikamine

Kõigepealt alustas kasutaja lõike printimisega. Seejärel teipis kasutaja prinditud lehed kokku. Peale seda suundus kasutaja veebirakenduse juurde tagasi ja lasi programmil välja arvutada lõike suurused. Vastavad arvutatud suurused otsis kasutaja lõikelt ülesse. Kõik suurused olid lihtsasti leitavad. Samuti erinevad värvid hõlbustasid kasutajal arusaada, millisest kohast tuleb lõigata. Lõike väljalõikamine kasutajale erilist segadust ei tekitanud ja kasutaja usub, et ka kooliõpilased suudaksid antud protsessi järgida.

6.1.3 Õpetuse jälgimine

Õpetus oli väga lihtsasti järgitav, kuid alles alustavale inimesele oleks võinud olla rohkem abistavaid pilte. Testkasutajale meeldis, et etapid olid jaotatud eri osadeks ja tekst ei olnud kuvatud kõik ühel lehel. Allääre töötlemise puhul oli toodud mitu eri valikuvõimalust, mis meeldis samuti kasutajale. Kirjelduste puhul soovitas testkasutaja lisada veel rohkem õpetusi, et muuta veel lihtsamaks arusaamise inimestele, kes ei ole kunagi õmblemisega tegelenud. Selle probleemi võib lahendada rohkemate piltide või videoklippide lisamine.

6.1.4 Kasutaja kokkuvõte

Kasutaja jäi väga rahule antud programmiga. Kõik tundus piisavalt selge ja läbimõeldud. Samuti toodi välja programmi erilisust, kuna sarnase rakendusega polnud kasutaja eelnevalt kokku puutunud. Valminud rakendust sooviks testkasutaja võtta kasutusele ka abistava vahendina tundides, et lihtsustada nii enda tööd, kui ka anda selgemat ülevaadet õpilastele.

6.2 Tulevikus loodavad arendused

Tulevikus on plaanis teha hulgaliselt edasiarendusi. Soov on lisada rakendusele kindlasti juurde erinevate keelte valikud, esialgselt inglise keele võimalus. Samuti jäi praegusest lõputöö skoobist välja makselahenduse loomine, kuna tulevikus on plaan luua ka tasulise

kasutaja versioon, milles on kasutajal suurem ligipääs keerukamate lõigete juurde ja samuti avada sama lõiget mitu korda, sisestades erinevaid mõõte. Hetkel on tavakasutajal lubatud vaid ühe korra sisestada mõõte ühe lõike ulatuses. Seega tasulise versiooni juures peaks kindlasti olema võimalus koheselt sooritada makse. Kasutaja mugavusi arvestades on plaanis teha ka sisselogimine võimalikuks läbi erinevate rakenduste. Kahjuks hetkel lõputöö aega selle arendamine ei mahtunud ja see tuleb teostada hilisema arenduse käigus.

Rakenduse peamine osa on sealsed lõiked ja nende õpetused. Kuna lõputöö aeg on üsna piiratud, siis hetkelises lahenduses on lisatud vaid kaks näidislõiget koos õpetustega, et anda aimdust kasutajale loodava rakenduse funktsionaalsusest. Tulevikus on plaan pidevalt lisada juurde uusi lõikeid ja nende õpetusi, juurde lisanduvad lõiked aitavad säilitada kasutajaskonda ja juurde meelitada uusi kasutajaid. Samuti võiks olla võimalus kasutajal saada teavitusi uute lõigete lisamisest.

Lisaks on tulevikus plaanis lisada kangakulu arvestamise süsteem. Loodavas süsteemis peaks kasutaja sisestama kanga laiuse, mille järel programm arvutab välja, kui palju on vaja kangast osta. See aitaks vähendada tekkivaid kangajäätmeid, mis säästab keskkonda. Samuti aitaks see hoida kokku raha, sest kangast pole vaja osta varuga.

7 Kokkuvõte

Lõputöö eesmärgiks oli luua veebirakendus, mis toetaks inimesi kogu õmblemisprotsessi ajal ja pakuks kasutajatele võimalust lasta programmil välja arvutada õmmeldava toote lõigete suurused. Lisaks oli vaja loodavas rakenduses pidada järge kasutaja läbitud juhenditest ja hetkel pooleliolevatest juhendite loetelust.

Analüüsi osas oli välja toodud tulevase veebirakenduse funktsionaalsed ja mittefunktsionaalsed nõuded. Samuti kasutajalugude abil kirjeldatud protsess, mis aitas paremini aru saada loodava rakenduse ülesehitusest. Analüüsi osas sai hea ülevaate võimalikest tehnoloogiatest veebirakenduse loomiseks. Arenduse peatükis toodi välja tulevase rakenduse teenuse- ja rakendusepoolne arhitektuur. Väljatoodud andmebaasi mudel andis hea ülevaate loodava rakenduse ülesehitusest.

Lõpptulemusena sai rakendusse lisatud kaks juhendit koos vastava lõigedega, mida on võimalik alla laadida. Kuna lõputöö aeg ja maht olid piiratud, siis kahjuks jäid välja mitmed funktsionaalsed nõuded ja samuti ei jõutud testimiseks lisada rohkem lõikeid ja õpetusi, mida kasutaja saaks proovida. Protsessi mitmeid kordi läbi proovides ja testides võib väita, et loodud rakendus vastas algselt püstitatud nõuetele ja peamine eesmärk sai saavutatud. Loodud rakendus on lihtsa ja selge struktuuriga, tulevikus uute lõigete ning õpetuste lisamine on muudetud väga lihtsaks.

8 Kasutatud kirjandus

[1] Northwestern Medicine, „Busting Body Type Workout Myths,“ jaanuar 2019. [Võrgumaterjal]. Available: https://www.nm.org/healthbeat/healthy-tips/fitness/busting-body-type-workout-myths. [Kasutatud 15 Veebruar 2022].

[2] Lectra, „Discover Modaris: our patternmaking software,“ [Võrgumaterjal]. Available: https://www.lectra.com/en/products/modaris-expert. [Kasutatud 15 Veebruar 2022].

[3] Tukatech, „Software for 2D CAD pattern making, grading and marker making,“ [Võrgumaterjal]. Available: https://tukatech.com/tukacad/. [Kasutatud 15 Veebruar 2022].

[4] Seamly, „About,“ [Võrgumaterjal]. Available: https://seamly2d.wordpress.com/about/. [Kasutatud 15 Veebruar 2022].

[5] P. Genevey, Rõivaste konstrueerimine ja õmblemine, Pariis: Edition Vigot, 2017.

[6] A. M. Menu, 8 Jaanuar 2021. [Võrgumaterjal]. Available: https://uxdesign.cc/5-reasons-why-you-should-not-have-business-logic-in-your-ui-layers-angular-wpf-why-not-to-go-36533ade6a84. [Kasutatud 17 Veebruar 2022].

[7] Async Labs, „5 critical reasons why you shouldn’t have business logic in your UI layer – why not to go for thick clients?,“ 21 Mai 2021. [Võrgumaterjal]. Available: https://www.asynclabs.co/blog/software-development/how-to-choose-the-right-backend-technology-for-your-app/. [Kasutatud 18 Veebruar 2022].

[8] A. Malhotra, „9 best reasons to choose java for web development,“ 22 Aprill 2019. [Võrgumaterjal]. Available: https://www.xicom.biz/blog/9-best-reasons-to-choose-java-for-web-development/. [Kasutatud 18 Veebruar 2022].

[9] A. Roznovsky, „Why use PHP? Main Advantages and disadvantages,“ [Võrgumaterjal]. Available: https://light-it.net/blog/why-use-php-main-advantages-and-disadvantages/. [Kasutatud 20 Veebruar 2022].

[10] A. Dziuba, „7 Advantages of Node.js for Startups,“ [Võrgumaterjal]. Available: https://relevant.software/blog/7-benefits-of-node-js-for-startups/. [Kasutatud 20 Veebruar 2022].

[11] J. Payne, „Benefits of C#,“ 5 Mai 2021. [Võrgumaterjal]. Available: https://www.codeguru.com/csharp/benefits-of-c/. [Kasutatud 20 Veebruar 2022].

[12] R. Krajewski, „C# vs Java: Which Is Better For Building Your Product?,“ 5 Juuli 2021. [Võrgumaterjal]. Available: https://www.ideamotive.co/blog/c-sharp-vs-java-which-is-better-for-building-your-product. [Kasutatud 25 Veebruar 2022].

[13] A. Aggarwal, „C# | .NET Framework (Basic Architecture and Component Stack),“ 19 Jaanuar 2019. [Võrgumaterjal]. Available: https://www.geeksforgeeks.org/c-sharp-net-framework-basic-architecture-component-stack/. [Kasutatud 1 Märts 2022].

[14] Microsoft, „What is .NET?,“ [Võrgumaterjal]. Available: https://dotnet.microsoft.com/en-us/learn/dotnet/what-is-dotnet. [Kasutatud 1 Märts 2022].

[15] N. Sakovich, „Top Most Popular Frontend Frameworks 2022,“ [Võrgumaterjal]. Available: https://www.sam-solutions.com/blog/best-frontend-framework/. [Kasutatud 5 Märts 2022].

[16] Angular, „What is Angular?,“ [Võrgumaterjal]. Available: https://angular.io/guide/what-is-angular. [Kasutatud 5 Märts 2022].

[17] TypeScript, „What is TypeScript?,“ [Võrgumaterjal]. Available: https://www.typescriptlang.org. [Kasutatud 6 Märts 2022].

[18] VueJS, „What is Vue?,“ [Võrgumaterjal]. Available: https://vuejs.org/guide/introduction.html. [Kasutatud 5 Märts 2022].

[19] M. Wolfe, „MySQL vs Oracle SQL,“ 20 August 2021. [Võrgumaterjal]. Available: https://towardsdatascience.com/mysql-vs-oracle-sql-a97a7659f992. [Kasutatud 9 Märts 2022].

[20] PostgreSQL, „About,“ [Võrgumaterjal]. Available: https://www.postgresql.org/about/. [Kasutatud 10 Märts 2022].

[21] B. Chen, „PostgreSQL vs. MySQL: What You Need to Know,“ 2 September 2021. [Võrgumaterjal]. Available: https://www.fivetran.com/blog/postgresql-vs-mysql. [Kasutatud 10 Märts 2022].

[22] Software testing help, „Difference Between SQL Vs MySQL Vs SQL Server (With Examples),“ 3 Aprill 2022. [Võrgumaterjal]. Available: https://www.softwaretestinghelp.com/sql-vs-mysql-vs-sql-server/. [Kasutatud 5 Aprill 2022].

[23] M. Metshein, „Git versioonihaldus,“ 23 August 2020. [Võrgumaterjal]. Available: https://www.metshein.com/git-versioonihaldus/. [Kasutatud 15 Märts 2022].

[24] M. Richards, Software Architecture patterns, California: O’Reilly Media, 2015.

[25] Microsoft, „Migrations Overview,“ 10 Oktoober 2021. [Võrgumaterjal]. Available: https://docs.microsoft.com/en-us/ef/core/managing-schemas/migrations/?tabs=dotnet-core-cli. [Kasutatud 20 Aprill 2022].

[26] Dotnet tutorials, „Unit Of Work in Repository Pattern,“ [Võrgumaterjal]. Available: https://dotnettutorials.net/lesson/unit-of-work-csharp-mvc/. [Kasutatud 20 Märts 2022].

[27] S. Mitchell, „Creating a Data Access Layer (C#),“ 2 Veebruar 2022. [Võrgumaterjal]. Available: https://docs.microsoft.com/en-us/aspnet/web-forms/overview/data-access/introduction/creating-a-data-access-layer-cs. [Kasutatud 20 Märts 2022].

[28] A. S. Gillis, „REST API (RESTful API),“ [Võrgumaterjal]. Available: https://www.techtarget.com/searchapparchitecture/definition/RESTful-API. [Kasutatud 21 Märts 2022].

[29] JSON, „Introducing JSON,“ [Võrgumaterjal]. Available: https://www.json.org/json-en.html. [Kasutatud 22 Märts 2022].

[30] MDN contributors, „Using FormData Objects,“ 28 Jaanuar 2022. [Võrgumaterjal]. Available: https://developer.mozilla.org/en-US/docs/Web/API/FormData/Using_FormData_Objects. [Kasutatud 20 Märts 2022].

[31] J. Ingeno, Software Architect's Handbook, Birmingham: Packt Publishing, 2018.

[32] V. Coisne, „What is Docker Compose? All you need to know!“ 3 Detsember 2020. [Võrgumaterjal]. Available: https://strapi.io/blog/what-is-docker-compose-all-you-need-to-know?msclkid=1cc9e255bbf611ec9a72ad449aa230fc. [Kasutatud 25 Märts 2022].

[33] Microsoft, „Entity Framework,“ 19 Veebruar 2020. [Võrgumaterjal]. Available: https://docs.microsoft.com/en-us/aspnet/entity-framework. [Kasutatud 1 Aprill 2022].

[34] Microsoft, „Language Integrated Query (LINQ) (C#),“ 18 Veebruar 2022. [Võrgumaterjal]. Available: https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/linq/. [Kasutatud 3 Aprill 2022].

[35] A. S, „A single-page application is a more modern approach to app development.,“ [Võrgumaterjal]. Available: https://lightrains.com/blogs/single-page-application-vs-multi-page-application/. [Kasutatud 5 Aprill 2022].

[36] ZetCode, „Axios tutorial,“ 18 Onktoober 2021. [Võrgumaterjal]. Available: https://zetcode.com/javascript/axios/. [Kasutatud 20 Aprill 2022].

[37] S. Curtis, 24 Aprill 2020. [Võrgumaterjal]. Available: https://stevenpcurtis.medium.com/endpoint-vs-api-ee96a91e88ca. [Kasutatud 21 Aprill 2022].

[38] T. Hawkins, 12 Juuli 2020. [Võrgumaterjal]. Available: https://levelup.gitconnected.com/protecting-against-xss-attacks-in-react-52442d9fff4c. [Kasutatud 21 Aprill 2022].

[39] G. K. Roy. [Võrgumaterjal]. Available: https://www.loginradius.com/blog/engineering/react-security-vulnerabilities/. [Kasutatud 21 Aprill 2022].

[40] Smartbear, „What Is Unit Testing?,“ [Võrgumaterjal]. Available: https://smartbear.com/learn/automated-testing/what-is-unit-testing/. [Kasutatud 10 Aprill 2022].

[41] M. Wolfe, „MySQL vs Oracle SQL,“ August 2021. [Võrgumaterjal]. Available: https://towardsdatascience.com/mysql-vs-oracle-sql-a97a7659f992.

[42] „Git versioonihaldus,“ 23 August 2020. [Võrgumaterjal]. Available: https://www.google.com/url?q=https://www.metshein.com/git-versioonihaldus/&sa=D&source=docs&ust=1650308635261349&usg=AOvVaw2popDLkCT3Ma98IVhlgKws.

[43] iFour Team, „Differences Between ASP.Net and ASP.Net Core - ASP.Net vs ASP.Net Core,“ 29 November 2019. [Võrgumaterjal]. Available: https://www.ifourtechnolab.com/blog/differences-between-asp-net-and-asp-net-core-asp-net-vs-asp-net-core. [Kasutatud 19 Aprill 2022].

[44] S. Nguyen, 16 Juuli 2020. [Võrgumaterjal]. Available: https://www.freecodecamp.org/news/what-is-npm-a-node-package-manager-tutorial-for-beginners/. [Kasutatud 20 Aprill 2022].

Lisa 1 – Valminud rakendus

Käesolev rakendus on ligipääsetav TTÜ IT Kolledži sisevõrgus

Rakenduse link: hlepik.itcollege.ee

Lisa 2– Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks[1]

Mina, Helen Lepiku

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose „Veebirakendus individuaalseks õmblemiseks kehamõõtude järgi“ , mille juhendajad on Meelis Antoi ja Andres Käver
   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.

24.04.2022

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