TARTU ÜLIKOOL
Arvutiteaduse Instituut
Informaatika õppekava

Madis Martin Lutter

Joogivanema infosüsteemi *front-end*'i prototüüp üliõpilasseltsile

Bakalaureusetöö (9 EAP)

Juhendaja: Anne Villems, MSc

Tartu 2016

Joogivanema infosüsteemi *front-end*'i prototüüp üliõpilasseltsile

Lühikokkuvõte: Selles töös antakse ülevaade joogivanema ülesannetest akadeemilises üliõpilasseltsis ning hetkel kasutatavast tehnilisest lahendusest, mis abistab joogivanemat tööülesannete täitmisel. Tuuakse välja hetkel kasutusel oleva süsteemi miinused ning töötatakse välja uus lahendus, mis lihtsustaks joogivanema tööd ning kõrvaldaks osa hetkel kasutusel oleva süsteemi kitsaskohtadest. Töö tulemusena valmib infosüsteemi rakendusliidese (API) spetsifikatsioon ning kasutajaliides, mis suhtleb planeeritava infosüsteemiga rakendusliidese kaudu.

Võtmesõnad: infosüsteem, veebirakendus, kasutajaliides, rakendusliides

CERCS: P175, informaatika, süsteemiteooria

Shop keepers information system front-end prototype for an academic organization

Abstract: In this thesis an overview of a shop keepers tasks in an academic organization and of the current technical solution used by the shop keeper is given. The problems with the current system are highlighted and a new and better solution for helping the shop keeper is developed. The result of this work is a specification for the application programming interface of the information system and an user interface which uses that interface for interacting with the planned information system.

Keywords: information system, web application, user interface, application programming interface, API

CERCS: P175, informatics, systems theory

Sisukord

Sissejuhatus

Üliõpilase jaoks on üks väärtuslikemaid ressursse aeg. Viimasel ajal üha karmistuv õppekorralduseeskiri jätab üliõpilastele õppetöövälisеks tegevuseks aina vähem aega. Selleks, et aega rohkem oleks, on üks võimalus aega kulutavaid ülesandeid automatiseerida. Akadeemilises üliõpilasseltsis EÜS Põhjala on üheks selliseks aega nõudvaks ülesandeks laohalduse, raamatupidamise ja aruandlusega seotud tegevused, mida sooritab üheks semestriks valitud ametikandja – joogivanem.

Töö eesmärk on välja töötada joogivanema infosüsteemi kasutajaliidese prototüüp ning rakendusliidese spetsifikatsioon infosüsteemile. Infosüsteemi eesmärk on lihtsustada joogivanema tööd, teda kontrolliva revisjoni tööd ning ühtlustada andmete salvestusviisi, võimaldades ülevaadet toimunud tehingutest läbi semestrite ühtsel kujul.

Esimeses peatükis kirjeldatakse joogivanema ja teda kontrolliva revisjoni tööülesandeid, heidetakse pilk hetkel kasutusel oleva tarkvara tugevustele ja nõrkustele ning arutletakse, mis probleemid lahendaks infosüsteemi välja arendamine. Teises peatükis antakse ülevaade sellest, mida infosüsteem peab võimaldama ja kes on infosüsteemi kasutajagrupid. Kolmandas peatükis on kirjeldatud planeeritava infosüsteemi ülesehitust. Neljanda peatüki alguses antakse juhised prototüübi oma arvutisse installeerimiseks ja jooksutamiseks. Seejärel kirjeldatakse töö käigus defineeritud rakendusliidese spetsifikatsiooni infosüsteemiga suhtlemiseks ja valminud kasutajaliidese prototüüpi.

1 Probleemi kirjeldus

Paljudes akadeemilistes üliõpilasorganisatsioonides pakutakse oma liikmetele võimalust organisatsiooni hoones süüa ja juua osta. Selleks, et toitu ja jooki jätkuks, valitakse üldjuhul organisatsiooni liikmete seast ametikandja, keda me nimetame selles töös joogivanemaks. Tema käsutusse antakse üheks semestriks rahalised vahendid toidu ja joogi soetamiseks. Semestri lõpus annab joogivanem ameti koos varadega üle järgmisele ametikandjale ning esitab aruande varadega seotud tehingute kohta, mis toimusid tema ametiaja jooksul. See võimaldab ülejäänud organisatsiooni liikmetel veenduda, et joogikassa varasid on kasutatud eesmärgipäraselt.

1.1 Joogivanema ameti kirjeldus

Esmapilgul ei pruugi joogivanema ülesanne näida kuigi keeruline. Teisalt võib joogivanema ametit võrrelda väikepoe omaniku tööga. Selleks, et oma tööd efektiivselt teha, on kummalgi tarvis infot laoseisu ja tarbimisharjumuste kohta. Paljude otsuste juures on oluline teada, kui palju on hetkel laos vaba ruumi, kui palju on vastavat kaubaartiklit laos, millal see rikneb, kui mitu ühikut keskmiselt päevas tarbitakse või mis hinnaga kaup sisse on ostetud. Erinevalt väiksest poest, tuleb hallata ka liikmete saldot joogikassas, juhul kui liikmetel lubatakse kaupa tarbida ettemaksust või jääda joogikassale võlgu. Joogikassaks nimetatakse kõiki varasid, mis on joogivanema hallata. Selleks on sularaha, raha pangakontol, laos olev kaup ja taara ning liikmete võlg joogikassa ees. Kirjeldame nüüd konkreetsemalt, mis olukordades mis infot joogivanemal tarvis on.

Otsustamaks, mis kaupa ja kui palju sisse osta, on tarvis teada, kui palju on laos vaba ruumi, kui palju on vastavat kaupa hetkel laos ja kui mitu ühikut vastavat toodet päevas keskmiselt tarbitakse. Tarbimiskiirus näitab ühest küljest, kas liikmetel on huvi vastava kaubaartikli vastu. Teisest küljest võimaldab see vältida olukordi, kus kiiresti riknevat kaupa ostetakse lattu sisse rohkem, kui ära jõutakse tarbida. Lisaks võimaldab tarbimiskiiruse ja laoseisu teadmine laopinda ära kasutada nii, et erinevad kaubaartiklid lõppevad otsa enam-vähem samal ajal. Selline planeerimine tähendab, et joogivanem ei pea nii tihti varusid täiendama, mis võimaldab aega kokku hoida.

Kauba edasimüügi hinna valimisel tuleb joogivanemal lähtuda kauba sisse ostmise hinnast ja mõnikord ka vastava kaubaartikli aegumiskuupäevast. Nähes, et mõni kaubaartikkel on lähiajal aegumas, saab joogivanem müügihinda langetada. See võib suurendada kauba tarbimise kiirust ning seeläbi saab vältida olukorda, kus palju kaupa tuleb aegumise tõttu maha kanda.

Akadeemilise organisatsiooni liikmeskond on kindlalt piiritletud võrreldes poe külastajatega. Seetõttu on tavaks organisatsiooni liikmetel lubada kaupa tarbida ettemaksust või jääda joogikassale võlgu. Joogivanemal on oluline teada, kes on võlgu ning kui kaua nad võlgu on olnud. Selle põhjal saab vajadusel piirata võlgu olevate liikmete tarbimist ning hinnata, kas võla tagasi nõudmisega on tarvis tegeleda ja kui ulatuslikke meetmeid selleks rakendama peaks.

Joogivanemal peab olema võimalik kontrollida, kas joogikassa varade eeldatav hulk vastab tegelikkusele. Varade eeldatavat hulka on võimalik välja arvutada esialgse varade koguste ning nendega sooritatud tehingute põhjal. Teades varade eeldatavaid koguseid saab joogivanem varad üle lugeda ning kontrollida, kas eeldatavad kogused klapivad te-

gelike kogustega. Juhul kui numbrid ei klapi viitab see kas sellele, et joogikassa varadega tehtud tehingud pole kõik üles märgitud või on keegi joogikassa varasid meelevaldselt omastanud.

Ametiaja lõppedes tuleb joogivanemal hinnata, kui suure summaga saab joogikassast toetada organisatsiooni järgmise semestri eelarvet. Selleks on tarvis teada joogikassa varade koguväärtust ning omada ülevaadet, kuidas see on semestri jooksul muutunud.

Lisaks eelnevale peab joogivanem olema ametiaja lõppedes aruandevõimeline. See tähendab, et tagantjärgi peab olema võimalik saada ülevaadet sellest, mis toimus joogikassale kuuluvate varadega joogivanema ametiaja jooksul.

1.2 Revisjonikomisjoni ameti kirjeldus

Kuna joogivanema käsutusse antakse organisatsioonile kuuluvad rahalised vahendid, on vajalik, et tema tegevust oleks võimalik ametiaja lõppedes kontrollida. Joogivanema tegevuse kontrollimiseks määratakse organisatsiooni liikmete seast inimesed, keda nimetame edaspidi revisjonikomisjoni liikmeteks või üldiselt revisjoniks.

Lühidalt kokkuvõetuna kontrollib revisjon:

1. kas vajalike tehingute jaoks on tšekid olemas;
2. kas arvutamisel on tehtud vigu;
3. kas pangakonto väljavõttes esinevad tehingud on kajastatud joogivanema tabelites;
4. kas tehtud kulutused on kooskõlas joogivanemale seatud eesmärkidega;
5. joogivanema hoolsust.

Tšekkide komplekt loetakse täielikuks siis, kui iga kauba lattu ostmise tehingu kohta on olemas tšekk. Samuti eeldatakse tšeki olemasolu nende tehingute kohta, mida loeme kuludeks. Kuludeks loetakse tehinguid, kus mingi joogikassa vara kogus väheneb, kuid vastu saadavat kaupa ei loeta joogikassa varaks (näiteks pastakate ostmine). Lisaks kontrollib revisjon, et tšekkidel olevad numbrid klapivad joogivanema tabelis olevate numbritega.

Revisjon kontrollib, kas joogivanem on teinud arvutus- või loogikavigu oma tabelites. Näiteks juhul kui joogivanem on arvutanud toodete omahinda, siis kas seda on tehtud korrektselt.

Revisjon võrdleb pangakonto väljavõtet joogivanema tabelitega ning kontrollib, kas iga pangakonto väljavõttel esinev tehing kajastub ka joogivanema tabelites. Näiteks võib esineda olukord, kus liige on sooritanud ettemaksu joogivanema pangakontole, kuid joogivanem pole seda märganud.

Revisjon hindab tehtud kulutuste põhjendatust. Näiteks, kui joogivanem ostab pastakaid, siis neid võib tõepoolest ametikohustuste täitmisel vaja minna. Kui joogivanem ostab joogikassa varade eest endale uue mobiiltelefoni, võib seda lugeda varade mitteeesmärgipäraseks kasutamiseks.

Revisjon hindab joogivanema hoolsust. Juhul kui kaupa on aegumiskuupäeva möödumise tõttu maha kantud, siis vaadatakse, kas joogivanem omas ülevaadet sellest, millal kaup rikneb, ja kas astuti ennetavaid samme selle vältimiseks.

Paneme tähele, et olukordades, kus joogikassa varade kogused ei klapi eeldatavate kogustega, pole võimalik revisjonil kindlaks teha, kas antud vara on varastatud joogivanema või mõne muu organisatsiooni liikme poolt.

1.3 Hetkel kasutatav süsteem

Joogivanema ameti täitmisega seotud info üles märkimine vajab põhjalikku läbimõtlemist. Seni on kasutatud selleks tabelarvutusprogramme, kuhu pannakse kirja kõik joogikassa varadega seotud tehingud. Algandmete põhjal on kõigile eespool mainitud küsimustele võimalik vastata. Lisades makrosid on võimalik luua isegi vaateid, mis arvutavad algandmete põhjal huvipakkuvaid tulemusi automaatselt, näiteks jooksvat laoseisu.

Tabelarvutusprogrammi kasutamine on üsna paindlik, kuna süsteemi on võimalik hõlpsasti jäädvustada kõikvõimalikke eriolukordi, lisades lahtritesse kommentaare või tabeleid ümber tehes. Teisalt võimaldab selline paindlikkus igal ametikandjal ise otsustada, kuidas ta andmed jäädvustab. Samuti nõuab joogikassa seisust ülevaate saamine ja tehingute jälgimine suurt pingutust teiste inimeste poolt, kes ei tööta joogivanema poolt välja töötatud tabelitega igapäevaselt ning ei näe tabelitevahelisi seoseid. Praktika on näidanud, et uuel ametikandjal on lihtsam välja töötada oma tabelite süsteem, kui ära õppida eelmise ametikandja süsteem ja sellega jätkata. Uue süsteemi välja töötamine on tudengi jaoks kindlasti arendav, kuid see on ka üsna aeganõudev ja veaohtlik. Tähelepanuta jäävad pisiasjad võivad tähendada, et semestri lõpus ei ole varade liikumine täielikult kontrollitav. Lisaks tähendab see lisatööd organisatsiooni liikmetele, kes soovivad teha statistikat joogikassa kohta üle mitme semestri. Andmed tuleb esmalt samale kujule viia, enne kui analüüsiga alustada saab.

1.4 Uus süsteem

Olukorras, kus kasutatakse tabelarvutusprogramme vajalike andmete jäädvustamiseks, meenutab joogivanema vahetumisega kaasnev tihtipeale jalgratta leiutamist. Ametisse sisse elamise lihtsustamiseks, arvutus- ja loogikavigade välistamiseks ning aruandluse ühtlustamiseks ja automatiseerimiseks tuleks välja arendada hõlpsasti õpitava kasutajaliidesega infosüsteem. Infosüsteem peab olema töökindel ja piisavalt paindlik, et olla võimeline modelleerima piisava täpsusega ka ettenägematuid olukordi. Vastasel juhul võib juhtuda, et tulevikus minnakse üle uuesti tabelarvutusprogrammide kasutamisele.

Infosüsteemi kasutamine võimaldaks tulevikus süsteemi edasi arendada, mis on välistatud hetkeolukorras, kus pea iga uus ametikandja loob uue süsteemi. Selleks tuleb infosüsteem välja arendada nii, et uutel liikmetel oleks lihtne selle ülesehitusest aru saada ning seda edasi arendada.

2 Nõuded infosüsteemile

Nõuete kirjeldamiseks kasutati agiilsest tarkvaraarendusest pärit kasutajalugude meetodit. Kasutajalood on laused, milles on defineeritud infosüsteemi kasutaja roll, soovitav funktsionaalsus ja põhjendus, miks tarkvara peab talle sellist tegevust võimaldama [1]. Näiteks:

"Joogivanemana tahan ma omada võimalust muuta toote müügihinda selleks, et aktsiisitõusu või inflatsiooni tingimustes joogikassa kahjumisse ei jääks või et peagi riknevale kaubale soodushinda määrata."

Selliselt kirja pandud nõudeid on arendajal võimalik prioritiseerida. Samuti võimaldab nõude vajalikkuse põhjendamine arendajal sügavamalt mõista tarkvara tellija vajadusi ning seeläbi pakkuda paremaid lahendusi probleemi lahendamiseks. Kuigi antud projektis on tarkvara tellija ja arendaja sama isik, võimaldab selline kirjeldamine siiski paremini arutleda, kuidas defineerida arendatava süsteemi võimaluste piirid piiratud ressursside tingimustes.

Lähtudes joogivanema ja revisjonikomisjoni ametitega seotud ülesannetest otsustati, et joogivanema infosüsteemi kasutajarollid on:

- Joogivanem
- Revisjon
- Tavaliige

Järgnevalt vaatleme iga kasutajarolliga seotud funktsionaalseid nõudeid. Kasutajalood on grupeeritud kasutajarolli järgi. Joogivanema rolliga seotud kasutajalood on:

1. Tahan teada, kui palju laos kaupa on, et otsustada, kas ja kui palju peaks kaupa juurde ostma.
2. Tahan teada, kui palju on hetkel laos vastavat toodet selleks, et otsustada kui palju vastavat kaupa juurde osta on vaja. Samuti võimaldab see mul kontrollida, kas tooteid on võetud ilma maksmata, lugedes üle kauba tegeliku koguse laos.
3. Tahan teada, millal rikneb laos olev kaup ja kui palju on vastava riknemiskuupäevaga kaupa, et otsustada kas on tarvis toote müügihinda vähendada, et toodet kiiremini tarbitaks.
4. Tahan omada võimalust muuta toote müügihinda selleks, et aktsiisitõusu või inflatsiooni tingimustes mitte kahjumisse jääda või et peagi riknevale kaubale soodushinda määrata.
5. Tahan teada, kui palju on kindla liikme ettemaks või võlg, et otsustada, kas ma müün talle toodet või mitte. Samuti tahan ma võlaga liikmetele nende võlgnevust meelde tuletada.
6. Tahan teada, kuidas on iga joogikassa vara kogus muutunud aja jooksul selleks, et välja arvutada joogikassa koguväärtus ja seeläbi otsustada, kui palju raha saab anda semestri lõpus järgmise semestri eelarve toetuseks.

7. Tahan omada võimalust sisestada süsteemi kõikvõimalikke tehinguid, mis muudavad mingi joogikassa vara kogust. See võimaldab mul semestri lõpus vältida eraldi aruande kirjutamist, sest kõik on süsteemis juba kirjas.

8. Tahan omada võimalust semestri alguses amet üle võtta selleks, et seada täpne ajaline piir, kus lõppeb eelmise joogivanema vastutus ja algab minu oma.

9. Tahan omada võimalust semestri lõpus amet üle anda selleks, et seada täpne ajaline piir, kus lõppeb minu vastutus ja algab järgmise joogivanema oma.

Revisjonikomisjoni liikme rolliga seotud kasutajalood:

1. Tahan teada, mis tehinguid joogivanem joogikassa varadega teinud on selleks, et kontrollida, kas varasid on kasutatud eesmärgipäraselt.

2. Tahan teada, kui muudetakse varasemat tehingut selleks, et omada ülevaadet sellest, kas midagi on muutunud.

3. Tahan teada, kui varasemalt kinnitatud inventuuri muudetakse selleks, et olla kursis joogivanema toimetustega.

4. Tahan teada, kui lisatakse tehing, mille kuupäev on varasem hiliseima kuupäevaga tehingust või inventuurist selleks, et mul ei jääks märkamata, kui muudetakse tehinguid, mille ma olen juba üle kontrollinud.

5. Tahan omada võimalust külmutada kõik tehingud minu poolt revideeritaval semestril selleks, et pärast aruande kinnitamist poleks võimalik andmeid enam muuta.

6. Tahan omada võimalust kinnitada tehinguid, mille tõenduseks on vajalik tšekki näidata selleks, et hiljem oleks ülevaade olemas sellest, milliste tehingute jaoks oli tšekk olemas ja milliste jaoks mitte.

Ülejäänud organisatsiooni liikmetega seotud kasutajalood:

1. Tahan teada, kui suur on minu ettemaks või võlg joogikassa ees selleks, et otsustada, kas mul on vaja raha juurde maksta või kas mul jätkub ettemaksu toote tarbimiseks.

2. Tahan teada, mis tehinguid joogivanem on teinud, sest see suurendab joogikassa läbipaistvust. Minu usaldust joogikassa vastu suurendab teadmine, kui suur osa toote eest makstavast summast läheb organisatsiooni toetuseks ja kui suure osa moodustab toote omahind.

Vaatleme nüüd infosüsteemile kehtestatud mittefunktsionaalsete nõuetega seotud kasutajalugusid. Nendeks on:

1. Joogivanemana tahan omada võimalust sisestada tehinguid süsteemi nutitelefonist, sest tehingu sisestamiseks on ebamugav iga kord arvutit käima panna.

2. Infosüsteem peab olema kasutatav Microsoft Windowsi, Apple OS X, Linuxi, Androidi ja iOS operatsioonisüsteemiga seadmetelt, sest erinevad inimesed kasutavad erinevaid seadmeid.

3 Arhitektuurilised valikud

Platvormisõltumatuse võimaldamiseks otsustati infosüsteemi prototüüp välja arendada veebirakendusena. Veebirakendust on võimalik kasutada iga seadmega, millel on veebibrauser.

Värkvõrgu (*Internet of Things*) kiire arengu tõttu [4] muutub aina tähtsamaks, et infosüsteemidega oleks mugav suhelda nii inimestel kui seadmetel. Seadmete jaoks on mugav infosüsteemiga suhelda otse, jättes vahele inimeste jaoks arendatud kasutajaliidese. Üks võimalus seadmetevahelise suhtluse võimaldamiseks on kasutada REST arhitektuuristiili toetavat APIt. REST on akronüüm inglisekeelsest terminist *REpresentational State Transfer* ning see on arhitektuuristiil, mille puhul liidesega suhtlemine toimub HTTP päringutega [2]. API on akronüüm inglisekeelsest terminist *Application Programming Interface*, mida on eesti keeles nimetatud rakendusliideseks [8]. See liides võimaldab rakendusega suhelda.

Seadmest, mis kasutaks REST tüüpi rakendusliidest, võib näite tuua külmiku kujul, mis kuvab organisatsiooni liikmele tema saldot joogikassas. Külmik saaks HTTP päringu abil vajamineva info REST tüüpi rakendusliideselt pärida.

Inimestel on infosüsteemi aga mugavam kasutada kenasti kujundatud kasutajaliidesega, mis võimaldab infosüsteemis olevaid andmeid muuta ja graafiliselt esitada. Kasutajaliidesega on võimalik kasutajat piirata, teavitades teda näiteks vigasest sisendist vormi väljas ning keeldudes selliste vormide *back-end*'ile saatmisest.

Infosüsteemi arendamist alustati REST API liidese defineerimisega. See võimaldab eraldada üksteisest täielikult info esituskihi ja info haldamisega seotud loogika. Kindla piiri seadmine liidese näol võimaldab infosüsteemi arendada modulaarselt. Leppides näiteks kokku, milline saab olema API liides on võimalik *front-end*'i ja *back-end*'i arendada üksteisest sõltumatult. Samuti on hiljem võimalik kogu *front-end* kui moodul välja vahetada uue vastu, muutmata sealjuures *back-end*'i. Lisaks on võimalik sama *back-end*'i kasutada ka teistel rakendustel.

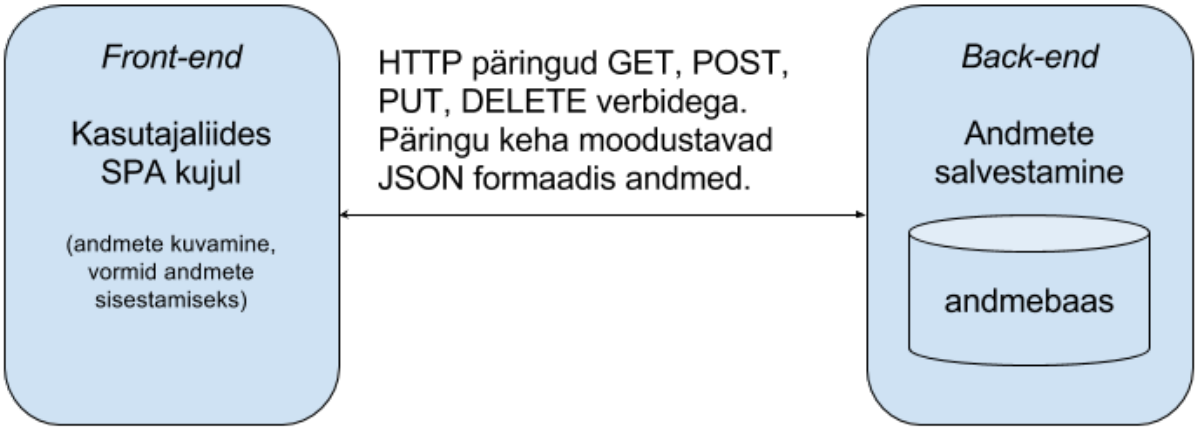
Joogivanema infosüsteem välja arendatud kujul jaguneb *front-* ja *back-end*'iks (vaata joonist 1). *Back-end* kujutab endast serverit, mis suhtleb andmebaasiga ja pakub REST tüüpi rakendusliidest. *Back-end*'ist võib mõelda kui andmesalvestuskihist, millele on lisatud loogikat ja mida on seetõttu mugavam kasutada.

Front-end kujutab endast veebirakendust, mis jookseb infosüsteemi kasutaja seadme brauseris. Antud projektis otsustati *front-end* realiseerida kui *single-page application* (SPA). SPA on veebirakendus, kus eri vaadete vahel navigeerides vahetatakse välja vaid osa veebilehe sisust – kordagi ei laeta kogu veebilehte uuesti kliendi brauserisse [5]. Tavaliselt laetakse vaate vahetamisel *back-end*'ist vaid vajalik info JSON või XML kujul. Eri vaadete kuvamiseks tarvilik HTML ning CSS laetakse kasutaja brauserisse tavaliselt lehe alglaadimisel, kuid kui lehe alglaadimise kiirust on vaja optimeerida, siis laetakse lehe alglaadimisel vaid vajaliku vaatega seotud HTML ja CSS. Ülejäänud vaated tuuakse alles siis, kui neid on vaja. *Single-page application* tüüpi lähenemine rakenduse ülesehitusele võimaldab vähendada serveri ja kliendi vahel liigutatavate andmete mahtu vaadete vahetamisel ja seeläbi muuta rakendust kiiremaks.

Suhtlus *front-* ja *back-end*'i vahel toimub HTTP päringutega. *Front-end*'ist saadetavad päringud kasutavad verbe GET, POST, PUT ja DELETE. Vastavalt vastuvõetud päringu verbile, aadressile, päisele ja kehale, sooritab *back-end* operatsioone andmebaasi ridadega.

See, mis operatsiooni teostada vastavalt millisele päringule on defineeritud REST tüüpi rakendusliidesega. Operatsioonid on ressursi loomine, lugemine, muutmine ja kustutamine. Inglisekeelses kirjanduses nimetatakse neid lühidalt CRUD operatsioonideks. CRUD on akronüüm sõnadest *Create, Read, Update, Destroy* [9]. Antud töös realiseeritud liidese puhul võib HTTP päringute verbid seada operatsioonidega vastavusse järgmiselt:

- POST - ressursi loomine (*Create*)
- GET - ressursi lugemine (*Read*)
- PUT - ressursi muutmine (*Update*)
- DELETE - ressursi kustutamine (*Destroy*)



Joonis 1: Joogivanema infosüsteemi ülesehitus. Rakendus jaguneb laias laastus kaheks: *front-end* ja *back-end*, mis suhtlevad omavahel HTTP päringutega.

HTTP päringutega masinloetava info vahetamiseks on kaks levinud levinud keelt, mida päringu kehas kasutatakse – JSON ja XML. Antud rakenduses otsustati *back-end*'iga suhtlemiseks kasutada HTTP päringute kehas JSON keelt, sest seda on lihtsam implementeerida ning see kogub aina rohkem populaarsust keele XML arvelt [3]. Töö autori subjektiivne arvamus on, et JSON on loetavam kui XML.

Front-end'i loomisel kasutati raamistikku AngularJS 1.5. SPA arendamist hõlbustavaid JavaScript keele raamistikke on olemas teisigi, näiteks Ember, Knockout, React ja Backbone [7]. Otsus langes AngularJS kasuks, kuna sellel on hea dokumentatsioon (vt `https://docs.angularjs.org/api`) ja see on teistest populaarsem [6], mis lihtsustab teistel arendajatel rakenduse lähtekoodi mõistmist ja projekti tulevikus edasi arendamist. Kasutajaliidese kujundamisel toetuti raamistikule Bootstrap 3.3 (vt `http://getbootstrap.com/`). Kasutatud ikoonid on pärit projektist Font Awesome 4.6 (vt `https://fortawesome.github.io/Font-Awesome/`). *Back-end*'iga suhtlemiseks kasutati teeki Restangular (vt `https://github.com/mgonto/restangular`). Kasutajale teavituste kuvamiseks kasutati teeki Toastr (vt `https://github.com/CodeSeven/toastr`). Need on vaid olulisemad teegid, millele toetuti. Ülejäänud teeke, millele toetuti, näeb lisas A olevas projekti repositooriumis projekti juurkaustas asuvast failist package.json.

4 Tulemus

Töö käigus töötati välja REST tüüpi API liides *back-end*'iga suhtlemiseks ning sellele liidesele toetuv *front-end*'i prototüüp, mille lähtekood on leitav lisast A. *Back-end* ise jäi implementeerimata, kuid töötati välja selle nõuded ning liides. *Front-end*'i arendamise hõlbustamiseks töötati välja *back-end* serveri *mock*, mis vastab GET päringutele failist loetud JSON andmetega. Selle lähtekood on leitav lisast B.

Prototüübi oma arvutis jooksutamiseks peab olema paigaldatud tarkvara *node* versioon 5.9.0 (vt `https://nodejs.org/en/`) ning selle tarkvaraga kaasa tulev paketihaldur *node package manager* (vt `https://www.npmjs.com/`). Varasemate versioonidega ei ole tarkvara testitud. Repositooriumi oma arvutisse kloonimiseks peab olema installeeritud tarkvara *git*. Prototüübi jooksutamiseks tuleb paigaldada nii prototüüp ise kui ka *back-end* serveri *mock*, kust *front-end* andmeid pärib.

Mock'i paigaldamiseks ja jooksutamiseks Linuxi operatsioonisüsteemiga arvutis tuleb navigeerida käsureal kausta, kuhu tarkvara soovitakse paigaldada ning jooksutada järgmised käsud:

```
git clone git@github.com:madislutter/jis-server.git
cd jis-server
npm install
node server.js
```

Viimane käsk käivitab *back-end* serveri, mis tuleb tööle jätta *front-end*'i kasutamiseks. *Front-end*'i paigaldamiseks ja jooksutamiseks tuleb avada uus käsurea sessioon ning jooksutada järgmised käsud:

```
git clone git@github.com:madislutter/jis-klient.git
cd jis-klient
npm install
npm install -g bower
bower install
npm install -g gulp
gulp serve
```

Viimane käsk käivitab *front-end*'i serveri ning avab brauseri navigeerides aadressile `http://localhost:3000`, kus rakendus asub. Järgnevalt antakse ülevaade välja arendatud tarkvarast.

4.1 *Back-end*'i REST API liides

Liidese spetsifikatsioon pandi töö käigus kirja *API Blueprint* formaadis (vt `https://apiblueprint.org/`). Selle faili põhjal genereeriti liidesele dokumentatsioon veebisaidi kujul tarkvara *aglio* abil (vt `https://github.com/danielgtaylor/aglio`). Parima ülevaate liidesest saab sirvides genereeritud dokumentatsiooni, mis on leitav lisast C.

Back-end'i pakutav liides defineerib järgmised ressursid:

- varad,
- tooted,
- kategooriad,

- liikmed,
- tehingud,
- inventuurid.

Varade ressurss võimaldab pärida joogikassa varade hulka soovitud ajahetkel. Joogikassa varadeks loetakse:

- laos olevate toodete omahindade summa,
- laos olev taara,
- liikmete saldode summa vastandväärtus,
- sularaha,
- arvel olev raha.

Samuti on võimalik pärida järjendit varade kogustest pärast iga tehingu sooritamist. See võimaldab kunagi joonistada graafikuid joogikassa varade väärtuste muutumisest läbi aja.

Toodete ressurss defineerib kõik tooted, mida laos võib esineda. Kategooriate ressurss näitab kõiki kategooriaid, mis toodetele on lisatud. Liikmete ressurss defineerib kõik organisatsiooni liikmed. Nii tooted kui liikmed on kustutatavad vaid siis, kui neile ei viita ükski tehing. Juhul kui neile viitab mõni tehing, saab neid kustutamise asemel arhiveerida. See on oluline sellepärast, et kui tehingult viidatavat objekti ei leidu, siis ei ole tehingust võimalik aru saada.

Pikemalt tuleb selgitada tehingute ressurssi. Tehingud on kõik joogikassa varadega sooritatud toimingud, mille käigus antud vara kogus muutub. Iga tehinguga on seotud kaks joogikassa vara. Üks varadest on tehingu kreeditpoolel, teine on tehingu deebetpoolel. Kreeditiks loetakse seda vara, mille kogus väheneb, deebetiks seda, mille kogus suureneb. Kreediti ja deebeti vara liigiks võib olla:

- toode,
- taara,
- liige,
- sularaha,
- pangakontol olev raha,
- tühi hulk.

Üldjuhul on kreedit ja deebet üksteisega võrdsed. Erandiks on tühja hulgaga tehingud ning ostu-müügitehingud, kus liikmele müüakse toodet. Sel puhul on deebeti summa võrdne müüdud toodete koguse ja edasimüügihinna korrutisega. Kreediti summa on aga võrdne müüdud toodete koguse ja omahinna korrutisega.

Tühi hulk võimaldab kujutada tehinguid, mille puhul joogikassa vara väheneb, kuid midagi vastu ei saada. Selliseks tehinguks loetakse kuluvahendite, näiteks pastakate, ostmine.

Samuti kasutatakse tühja hulka n-ö parandustehingutes, mida selgitatakse inventuuride juures.

Iga uue tehingu lisamisel või mõne olemasoleva tehingu muutmisel kontrollib süsteem, kas antud tehingu või mõne sellele ajaliselt järgneva tehingu järgselt muutub mõne joogikassa vara kogus negatiivseks. Tehingud, mille korral nii juhtub, märgitakse illegaalseteks. See abistab joogivanemat, juhtides tema tähelepanu sellele, et kuskil võib olla tehtud viga tehingute süsteemi kandmisel.

Inventuurid kujutavad joogikassa varade ülelugemise tulemusi. Joogikassa varasid on tarvis üle lugeda veendumaks, et midagi pole varastatud ning et kõik sooritatud tehingud on korrektselt süsteemi sisse kantud. Inventuur sisaldab infot üle loetud toodete, taara, sularaha ja arvel oleva raha koguse kohta. Inventuuri objektis üle loetud tooted peavad hõlmama kõiki viimases inventuuris nullist erineva kogusega tooteid ning kõiki tooteid, mida on pärast viimast inventuuri lattu kantud. See tähendab, et üle loetakse ka tooted, mille eeldatav kogus on null. Lisaks salvestab süsteem inventuuri objekti külge automaatselt ka liikmete saldode summa vastandväärtuse inventuuri hetkel. Inventuuri sooritaja ei saa seda kuidagi kokku lugeda. See suurus on vaid varasemate tehingute põhjal välja arvutatav.

Juhul kui inventuuril loetud varade kogused ei klapi eeldatavate kogustega, lisab süsteem automaatselt n-ö parandustehingud tühja hulga ja vastava vara vahel, mis viivad varade hulga vastavusse üle loetud kogustega. Hiljem, kui inventuuri või mõnda tehingut enne vastavat inventuuri muudetakse, muudetakse ka vastava inventuuriga seotud parandustehinguid.

Toome selle illustreerimiseks näite. Inventuuril loetakse sularaha koguseks 100 €, kuid vastavalt sularaha kogusele ameti üle võtmisel ning sellele järgnenud tehingutele pidanuks sularaha olema inventuuri sooritamise hetkel hoopis 120 €. Pärast inventuuri kinnitamist lisab süsteem automaatselt tehingu, mille kreeditpoolel on 20 € sularaha ning deebetpoolel tühi hulk. Seejärel meenub joogivanemale, et enne inventuuri maksis ta ühele liikmele tema ettemaksu tagasi 20 € väärtuses, kuid unustas vastava tehingu süsteemi lisada. Nüüd lisab ta inventuurieelsesse perioodi tehingu, mille kreeditpoolel on 20 € sularaha ning deebetpoolel 20 € vastava liikme krediiti. Pärast selle tehingu lisamist kustutab süsteem automaatselt viimase inventuuriga seotud parandustehingu sularaha ja tühja hulga vahel.

Tabelis 1 on näidatud kasutajarollide õigusi erinevate liidese ressurssidega. Tavaliikmed saavad kõiki ressursse näha, kuid ei saa midagi muuta. Revisjon saab lisaks kõigi ressursside nägemisele tehinguid muuta, kuid vaid nii palju, et märkida neid kinnitatuks. Joogivanem saab oma ametiajal muuta kõiki ressursse. Erandiks on tehingud. Tehingute muutmise õigus on seotud tehingu sooritamise hetkega, mitte sellega, kas joogivanem on päringu saatmise hetkel ametis. See on vajalik võimaldamaks joogivanemale tehingute muutmist vigade parandamise eesmärgil pärast ameti üle andmist järgmisele liikmele. See-eest ei või ta muuta uue ametikandja tehinguid ega lisada tehinguid perioodi, mil ta polnud joogivanem.

Parandustehingute puhul tohib joogivanem muuta vaid tehingu selgitust, sest nende tehingute kreedit ja deebet sõltuvad ülejäänud tehingutest ja inventuuridest.

Kuna joogivanem tohib muuta oma ametiaja tehinguid ka pärast oma ametiaja lõppu, peab revisjonil olema võimalus mingil hetkel külmutada kõik revideeritava semestri tehingud. Vastasel juhul võib juhtuda, et joogivanem muudab tehinguid, mida revisjon on

Päring		Roll		
Aadress	**Verb**	**Liige**	**Revisjon**	**Joogivanem**
/varad	GET	+	+	+
/tooted	GET	+	+	+
	POST	-	-	+
/tooted/id	GET	+	+	+
	PUT	-	-	+
	DELETE	-	-	+
/liikmed	GET	+	+	+
	POST	-	-	+
/liikmed/id	GET	+	+	+
	PUT	-	-	+
	DELETE	-	-	+
/tehingud	GET	+	+	+
	POST	-	-	+
/tehingud/id	GET	+	+	+
	PUT	-	+	+
	DELETE	-	-	+
/inventuurid	GET	+	+	+
	POST	-	-	+
/inventuurid/id	GET	+	+	+
	PUT	-	-	+
	DELETE	-	-	+

Tabel 1: Kasutajarollide õigused erinevate ressurssidega. + tähistab õiguste olemasolu vastavas rollis, − näitab, et vastavas rollis kasutajal õigusi ei ole.

juba kontrollinud.

Semestri algus- ja lõpphetk peavad olema süsteemis defineeritud ning seotud inventuuridega, sest joogikassa varade hulk tuleb fikseerida ameti üle andmisel. Kuidas määrata semestri algus- ja lõpphetke, vastaval semestril ametis olevaid joogivanemaid ja revisjonikomisjoni liikmeid, kuidas realiseerida tehingute külmutamine ning kuidas süsteemi kasutajaid autentida on jäetud lahtiseks.

4.2 *Front-end*'i prototüüp

Front-end'i prototüübi vormide kujundamisel on tähelepanu pööratud sellele, et kasutajat võimalikult palju juhendada andmete sisestamisel. Juhul kui sisestusvälja sisestatud andmed ei vasta oodatule, juhitakse sellele kasutaja tähelepanu ning keeldutakse vormi *back-end*'ile saatmisest. Nii on viidud miinimumini võimalus, et rakenduse kasutaja saaks kasutajaliidesest saata *back-end*'ile päringut, mida ei aktsepteerita. Tänu sellele näeb kasutaja oma viga andmete sisestamisel juba vormi täitmise ajal ja saab vea kohe parandada. Tuleb silmas pidada, et pahatahtlikud kasutajad saavad *back-end*'ile saata päringuid, mille kehas on nende enda koostatud sisu. Seega peab *back-end* kindlasti ka ise valideerima talle saadetud päringutes olevaid andmeid.

Rakenduse eri vaadete disainimisel on püütud sarnase funktsionaalsusega nupud paiguta-

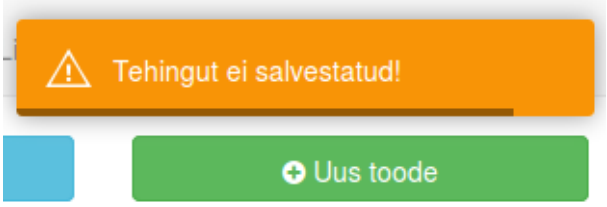
da samasse kohta. See hõlbustab rakenduse kasutajal vajaliku funktsionaalsuse leidmist. Näiteks on uue ressursi loomise nupp igas vaates üleval paremal nurgas ning otsingukast üleval vasakus nurgas.

Värvide kasutamisel on läbivalt lähtutud järgmistest põhimõtetest:

- Roheline värv tähistab ohutuid tegevusi ja tegevuste õnnestumist.
- Punane värv tähistab pöördumatuid tegevusi ja süsteemi vigasid. Pöördumatu tegevus on näiteks ressursi kustutamine, sest seda pole võimalik hiljem taastada.
- Kollane värv tähistab hoiatusi ja pooleli jäetud tegevusi.

Back-end liidese ressurssidega manipuleerimise õnnestumise, ebaõnnestumise ja pooleli jätmise kohta näidatakse kasutajale teavitusi ekraani ülal-paremal nurgas. See annab kasutajale kindluse tegevuse õnnestumise korral ning informeerib teda juhtudel, kus midagi on valesti. Näidet teavitusest näeb jooniselt 2.

Veebirakenduse menüüreal, mis on igas vaates nähtaval, kuvatakse joogikassa varade hetkeseisu. Iga vara tüübi jaoks kasutatakse erinevat ikooni. Viies kursori vastava vara kohale ilmub *tooltip*, mis selgitab, mis vara kogust antud number näitab. See aitab kasutajal õppida, mis ikoon mis varale vastab. Samu ikoone kasutatakse samade varade tähistamiseks kogu rakenduses. Näidet varade seisu kuvamisest näeb jooniselt 3.



Joonis 2: Ekraani ülal-paremas nurgas kasutajale kuvatav teavitus. Kasutajale antakse märku, et tegevus jäeti pooleli.

Joonis 3: Menüüreal kuvatakse kasutajale joogikassa varade hetkeseisu. Iga vara tähistab oma ikoon. Ikoonide tähenduse õppimise hõlbustamiseks on lisatud *tooltip* igale varale. Väikse ekraaniga seadmetel on varade hetkeseis ekraaniruumi kokku hoidmiseks peidetud.

Front-end prototüübil on neli vaadet:

1. toodete vaade,
2. liikmete vaade,
3. tehingute vaade,
4. inventuuride vaade.

Võimaldamaks infosüsteemi mugavalt nutitelefonist kasutada, on väiksema ekraaniga seadmetes osa infost peidetud. Selleks kaaluti, mis info nutitelefoniga infosüsteemi kasutajale vähemoluline on. Näiteks peidetakse toodete vaates väikse ekraaniga seadmes toote

omahind, müügihind ja lähim aegumiskuupäev. Samuti on peidetud toote kustutamise, arhiveerimise või ennistamise nupp. Menüürealt peidetakse väiksel ekraanil varade seis ning eri vaadete vahel navigeerimise nuppude asemel näidatakse kasutajale menüünuppu, millele klikkides näidatakse alles eri vaadetesse navigeerimist võimaldavaid nuppe. See tähendab küll, et teise vaatesse navigeerimiseks peab kasutaja ühe kliki rohkem tegema, kuid antud olukorras on ekraaniruumi kokku hoidmine tähtsam.

4.2.1 Toodete vaade

Toodete vaates kuvatakse süsteemis olevaid tooteid tabeli ridadena. Iga toote real näidatakse selle nime, taarat, ühiku kogust, omahinda, edasimüügihinda, laoseisu, lähimat aegumiskuupäeva ja vastava aegumiskuupäevaga toodete laoseisu. Lisaks on iga toote real nupud vastava toote lattu kandmiseks, maha kandmiseks, toote info muutmiseks ning toote kustutamiseks, arhiveerimiseks või ennistamiseks vastavalt toote kustutatuse ja arhiveerituse staatusele.

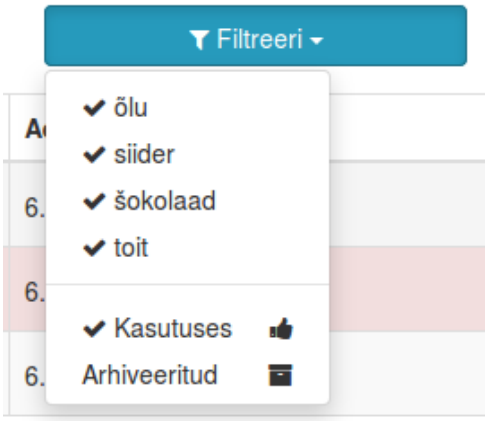
Toote arhiveerimine on keelatud, kui toote laoseis on nullist erinev. Ekraaniruumi kokku hoidmiseks on igal nupul vaid ikoon, mis viitab nupuga seotud tegevusele. Lisaks on igale nupule lisatud *tooltip*, kus on kirjas nupuga seotud tegevus. Sama lähenemist on kasutatud ka ülejäänud vaadetes nuppudega, mis võimaldavad tabeli real kujutatud ressursiga seotud toiminguid. Näidet toodete vaatest mobiilil näeb jooniselt 4.

Joonis 4: Toodete vaade mobiilil. Mobiilses vaates on toote omahind, edasimüügihind ja aegumiskuupäev peidetud, et ekraaniruumi kokku hoida. Samuti on peidetud nupp toote kustutamiseks või arhiveerimiseks. Peagi aeguva toote taust on punasega tähistatud.

Juhtimaks joogivanema tähelepanu peagi riknevale kaubale, on toote rida punaka taustaga siis, kui toote aegumiskuupäev on lähemal kui üks kuu või kui toode on juba aegunud.

Toodete leidmise hõlbustamiseks saab tooteid filtreerida otsingusõna, kategooria ja arhiveerituse staatuse järgi. Toodete filtreerimine käib rippmenüüst, millest näeb näidet

jooniselt 5. Lisaks on võimalik tooteid sorteerida nime, laoseisu, omahinna, edasimüügihinna ja lähima aegumiskuupäeva järgi.



Joonis 5: Toodete filtreerimine. Filtreerida saab kategooria ning arhiveerituse järgi.

Toote andmete muutmine toimub dialoogiaknas, mis avaneb pliiatsi ikooniga nupule klikkides vastava toote real. Dialoogiaknas kuvatakse kasutajale vormi, kust ta saab muuta toote nime, taara nimetust, ühiku kogust, müügihinda, triipkoodi ja kategooriaid. Kasutajal ei lubata toote andmeid salvestada, kui kasvõi üks sisestusväli ei vasta seatud nõuetele. Nõueteks on näiteks see, et tootel peab olema nimi ning müügihind. Müügihind peab sealjuures olema mittenegatiivne arv, mis on defineeritud mitte rohkem kui kahe komakoha täpsusega. Taarat omavalt tootelt ei lubata taarat eemaldada ning taarata toote puhul on taara nimetuse sisestusväli peidetud. See lihtsustab *back-end*'i loogikat. Juhul kui kasutajat ei piirataks selles osas, tuleks välja mõelda, mida teha, kui laos olevalt tootelt taara eemaldatakse. Kas süsteem peaks tootelt taara eemaldamise korral lisama automaatselt tehingud, mis kannavad antud toote laoseisule vastaval määral taarat maha? Või tuleb lihtsalt edasiste tehingute juures arvestada, et toodet sisse ostes pole vastaval määral taarat tarvis lattu kanda? Neile küsimustele vastamiseks oleks tarvis näiteolukorda, mis puhul kasutajal on tarvis tootelt taara eemaldada. Kuna töö autor ei suutnud sellist olukorda ette kujutada, otsustati see võimalus keelata.

Uue toote loomine toimub dialoogiaknas, mis avaneb nupule "Uus toode" klikkides. Avanev dialoogiaken on peaaegu identne toote info muutmise dialoogiaknaga. Vältimaks võimalikke segadusi on dialoogiakna pealkiri siiski erinev. Lisaks on infosüsteemi kasutaja juhendamiseks sisendväljadel kasutatud HTML *input* välja atribuuti *placeholder*. See kuvab kasutajale näidet sellest, millist väärtust antud sisendvälja oodatakse. Näide kaob, kui kasutaja sisestusväljale midagi sisestab.

Toote sisse ostmise nupul vajutamine avab dialoogiakna, kus kuvatakse kasutajale vormi. Antud vormis on kasutaja eest toode juba valitud ning seda muuta ei saa. Sisestada tuleb kauba eest makstud kogusumma, kauba kogus ja kauba aegumiskuupäev. Kauba tükihind arvutatakse kasutaja eest ning seda muuta ei saa. See on oluline sellepärast, et mõnikord ei jagu toote ühiku hind sendi täpsusega. Sellisel juhul peaks kasutama hakkama mõistatama, kui mitme komakohaga kujutada toote tükihinda. Selle asemel sisestab kasutaja vaid koguhinna ning koguse ja tükihind arvutatakse kasutaja eest nii täpselt, kui süsteem seda kujutada võimaldab. Kasutaja peab valima, kas maksis toote eest sularaha või kaardiga. Juhul kui lisatava tehingu tootel on taara, tehakse *back-end*'ile lisapäring, millega lisatakse vastava koguse taara sisseostu tehing samuti süsteemi.

Toote maha kandmise nupul vajutamine avab samuti dialoogiakna, kus kuvatakse kasutajale vormi, milles toode on juba valitud. Kasutaja peab sisestama, kui mitu toodet maha kanda ning põhjendama, miks toodet maha kantakse. Juhul kui tootel oli taara, peab kasutaja märkima, kas toote taara jäi terveks või tuleb ka see maha kanda. Kasutajale kuvatakse, kui suur kahju kaasneb vastava koguse toote maha kandmisega.

4.2.2 Liikmete vaade

Liikmete vaates kuvatakse süsteemis olevaid liikmeid tabeliridadena. Kuvatakse iga liikme nime ja saldot. Võimaldamaks visuaalsemat ülevaadet, on liikme rida punase taustaga, kui tema saldo on miinuses, ning rohelise taustaga, kui tema saldo on plussis. Lisaks on iga liikme real nupud talle ettemaksu lisamiseks, tarbimise lisamiseks, tema andmete muutmiseks ja liikme süsteemist kustutamiseks, arhiveerimiseks või ennistamiseks vastavalt liikme kustutatavuse ja arhiveerituse staatusele. Näidet liikmete vaatest näeb jooniselt 6.

Joonis 6: Liikmete vaade mobiilil. Võlaga liikmed on punase taustaga, ettemaksuga liikmed on rohelise taustaga. Ülejäänud liikmete taust on valge. Liikmete arhiveerimise nupp on peidetud ekraaniruumi kokku hoidmiseks väikse ekraaniga seadmetel.

Liikme arhiveerimise korral kustutatakse tema ettemaks või võlgnevus joogikassa ees. See on pöördumatu protsess ning seetõttu on oluline vähendada tõenäosust, et infosüsteemi kasutaja arhiveerib liikme kogemata. Selle vältimiseks kuvatakse süsteemi kasutajale liikme arhiveerimise nupule klikkides dialoogiakent, mille sisu sõltub liikme saldost. Juhul kui liikme saldo on nullis, küsitakse kasutajalt vaid kinnitust. Juhul kui liikme saldo on nullist erinev, teavitatakse kasutajat, et liikme arhiveerimise korral tema võlg või ettemaks joogikassas kustutatakse. Sel puhul peab infosüsteemi kasutaja kirjutama vastava liikme nime sisestuskasti. Alles seejärel lubatakse kasutajal kinnitada oma tegevus. Näidet kinnitusdialoogist näeb jooniselt 7.

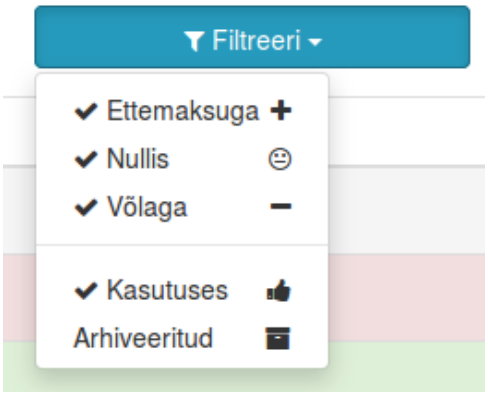
Liikmete otsimise hõlbustamiseks on võimaldatud liikmete filtreerimine otsisõna järgi, selle järgi kas liikme saldo on plussis, miinuses või nullis ning liikmete arhiveerituse staatuse järgi. Näidet filtreerimise rippmenüüst näeb jooniselt 8.

Liikmega seotud info muutmine toimub dialoogiaknas, mis avaneb pliiatsi ikooniga nupule vajutades vastava liikme real. Dialoogiaknas kuvatakse kasutajale vormi, kus tal on

Joonis 7: Liikme arhiveerimise kinnitusdialoog. Kasutajat hoiatatakse enne liikme arhiveerimist, et liikme ettemaks kustutatakse. Selleks, et kasutaja saaks vajutada nupule "Arhiveeri liige" tuleb kirjutada liikme nimi sisestuskasti. Ohtliku tegevuse märgistamiseks on tegevust käivitav nupp punast värvi, erinevalt tehingute sisestamise dialoogidest, kus see on roheline.

võimalik muuta liikme nime, koondist, meiliaadressi ja telefoninumbrit. Liikme andmeid ei lubata salvestada, kui liikmele ei ole defineeritud nime või kui sisestatud meiliaadress ei ole korrektne. Uue liikme loomise dialoog on peaaegu identne liikme info muutmise dialoogiga. Selle pealkiri on erinev ning kasutajatele kuvatakse tühjades sisestusväljades näiteandmeid.

Kasutajale ettemaksu lisamine või selle tagastamine toimub dialoogiaknas, mis avaneb pluss märgiga nupule klikkides vastava liikme real. Liikme lahter on kasutaja eest täidetud ning seda muuta ei saa. Kasutajalt küsitakse, kui palju liikmele ettemaksu lisada ning kas raha laekus ülekandega või anti sularahas. Juhul, kui kasutaja sisestab summa lahtrisse negatiivse arvu, vahetab *front-end back-end*'ile saadetavas tehingu loomise päringus kreediti ja deebeti omavahel ära, võimaldades samast vormist ka ettemaksu liikmele tagastada. Summa lahtrisse sisestatud number peab olema nullist erinev arv, mis pole defineeritud täpsemalt kui kaks komakohta.



Joonis 8: Liikmete filtreerimise rippmenüü. Välja saab filtreerida liikmed, kellel on võlg, nullsaldo või ettemaks. Samuti saab filtreerida arhiveerituse staatuse järgi.

Kasutajale tarbimise lisamine toimub dialoogiaknas, mis avaneb ostukäru ikooniga nupule klikkides vastava liikme real. Liikme lahter on kasutaja eest täidetud ning seda muuta ei saa. Kasutaja peab määrama, mis toodet kasutajale müüakse, kui palju ning kas müüakse vastava liikme ettemaksust või maksab liige toote eest sularahas. Toote valimise hõlbustamiseks kuvatakse kasutajale kõiki võimalikke tooteid, mille nimi klapib vormi sisestatud tekstiga. Kuna üldjuhul ostetakse üks toode korraga, on koguse lahter eeltäidetud numbriga 1. Juhul kui valitud tootel on taara, küsitakse kasutajalt, kas toode müüakse liikmele koos taaraga. Üldju-

hul jääb taara joogikassale, seega on vastav valik vaikimisi aktiivne. Üldlevinud valikute eeltäitmine võimaldab tehingu sisestamisel infosüsteemi kasutaja aega kokku hoida.

4.2.3 Tehingute vaade

Tehingute vaates kuvatakse joogikassa varadega tehtud tehinguid tabeli ridadena. Iga tehingu kohta kuvatakse kreedit- ja deebetpoole vara liiki, summat, tehinguga seotud selgitust ning tehingu toimumise hetke. Varade liike kuvatakse ikoonidega, hoidmaks kokku ekraaniruumi. Juhul kui vara liik on toode või liige, siis on lisatud ikoonile *tooltip*, millel näidatakse, mis toote või liikmega oli tegu. Näidet sellest näeb jooniselt 9. Lisaks on iga tehingu real nupud vastava tehingu muutmiseks ning kustutamiseks.

Parema ülevaatlikkuse võimaldamiseks värvitakse tehingu taust roheliseks, kui tegu oli kasumliku tehinguga, ning punaseks, kui tegu oli kahjumliku tehinguga. Süsteemi poolt lisatud parandustehingute taust värvitakse siniseks ning illegaalsete tehingute taust kollaseks.

Joonis 9: Tehingu liigi kuvamine. Kreedit on vasakul pool ning deebet paremal pool. Nool näitab varade teisenduse suunda. Toote *tooltip* näitab, et tehingu kreeditpoolel oli "Milka Daim".

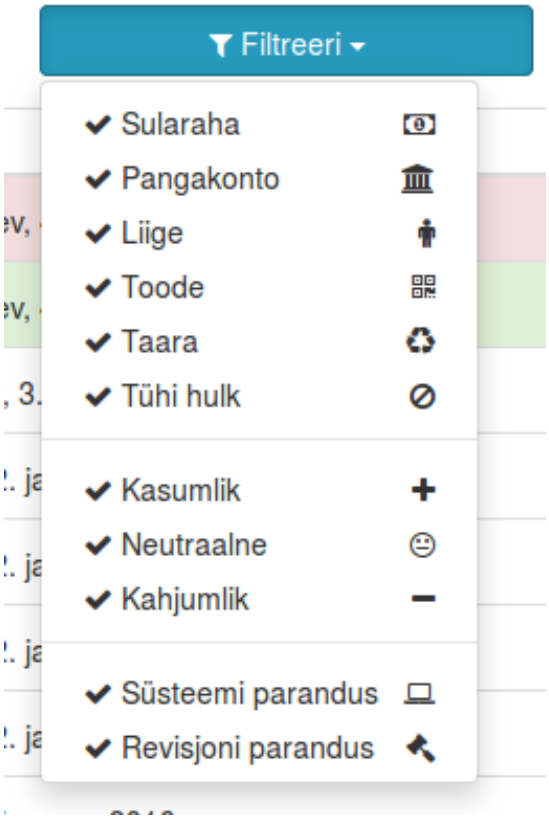
Erinevalt toodetest ja liikmetest, pole tehingute arhiveerimist tarvis võimaldada, sest tehingutele ei viita ükski teine objekt infosüsteemis. Kuna tehingu kustutamine on pöördumatu tegevus, siis küsitakse kasutajalt kinnitust enne tehingu kustutamist. See erineb toodete ja liikmete kustutamisest, mille puhul kinnitust ei küsita. Seda seetõttu, et tooteid ja liikmeid on võimalik kustutada vaid seni, kuni neile ei viita ükski tehing. Pärast liikme või toote süsteemi lisamist hakkab neile üsna peagi mõni tehing viitama, seega on võimalus, et toode või liige tahtmatult kustutatakse üsna väike. Samuti on toode või liige võimalik üsna hõlpsasti uuesti luua pärast tahtmatut kustutamist, sest objektidega seotud andmed on reaalsest elust kättesaadavad. Tahtmatult kustutatud tehinguga seotud andmeid ei pruugi tagantjärgi olla võimalik välja uurida, eriti sularahatehingute puhul.

Soovitud tehingute leidmise hõlbustamiseks saab neid filtreerida järgmiste parameetrite alusel:

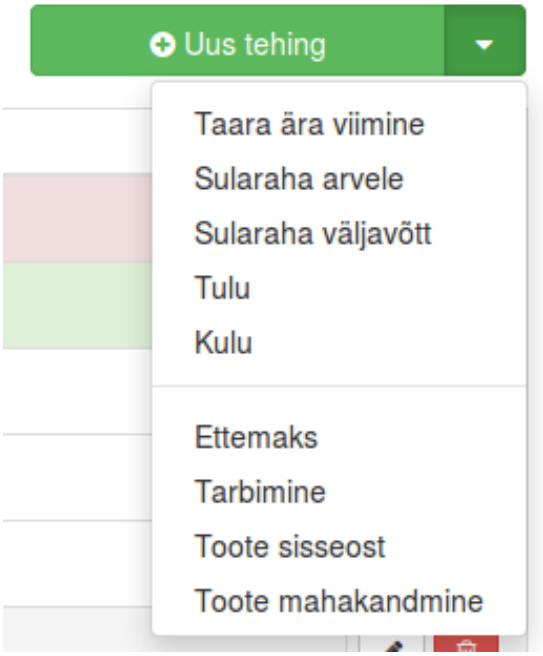
- tehingu toimumise hetk,
- tehinguga seotud varade liik,
- tehingu kasumlikkus, kahjumlikkus või neutraalsus joogikassa koguväärtusele,
- kas tegu on kasutaja lisatud tehingu või süsteemi lisatud parandustehinguga.

Samuti on võimalik tehinguid filtreerida otsisõna järgi. See võimaldab näiteks otsida tehingut selgituse ja tehingu kreedit- ning deebetpoole summa järgi. Näidet tehingu filtreerimise rippmenüüst näeb jooniselt 10. Tehinguid saab sorteerida tähestikuliselt selgituse järgi ning tehingu toimumise aja järgi.

Tehingute süsteemi lisamiseks on kaks moodust. Toodete ja liikmete vaates selgitatud toote lattu kandmise, maha kandmise, liikmele ettemaksu ja tarbimise lisamise tehinguid



Joonis 10: Tehingute filtreerimine. Filtreerida saab aja järgi, tehingu poolte järgi, selle järgi, kuidas tehing mõjutas joogikassa koguväärtust, selle järgi, kas tehing on loodud süsteemi poolt või mitte.



Joonis 11: Enamlevinud tehingud, mille jaoks on loodud dialoogid. Toodete ja liikmetega seotud tehingute lisamine on võimalik nii tehingute vaatest kui ka toodete ja liikmete vaadetest.

lisatakse spetsiaalselt vastavat tüüpi tehingu jaoks välja arendatud dialoogiga. Kasutatakse ära teadmist, mis tehingut kasutaja süsteemi sisestada tahab ning küsitakse vaid vajalikku infot vastava tehingu lisamiseks. Lisaks eelmainitud toodete ja liikmetega seotud tehingutele on välja arendatud eridialoogid ka taara ära viimise, sularaha arvele panemise, sularaha väljavõtmise, tulu ja kulu tüüpi tehingute jaoks. Näidet rippmenüüst, kust saab eelnimetatud tehinguid lisada, näeb jooniselt 11.

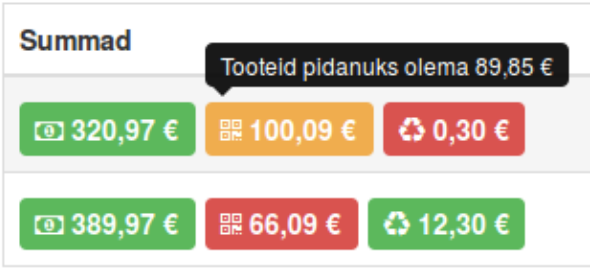
Juhul kui süsteem võimaldaks kasutajal süsteemi sisestada vaid eelmainitud tehinguid, võib juhtuda, et süsteem ei suuda reaalsust piisavalt hästi modelleerida. On võimalik, et tulevikus tuleb infosüsteemis kajastada sellist tüüpi tehingut, mida tarkvara arendaja ette ei näinud. Selleks, et infosüsteem oleks võimalikult paindlik, on võimaldatud kasutajal lisada tehinguid süsteemi nii, et tehingu kreedit ja deebet määratakse käsitsi. Nii on võimalik sooritada tehingut ükskõik millise kahe joogikassa vara vahel. Näiteks saab liige taara eest endale ettemaksu osta. Sellisel juhul oleks tehingu kreeditiks liige (liikme võlg joogikassa ees väheneb) ja deebetiks taara (taara kogus suureneb).

Selliselt tehingute süsteemi lisamine on veaohtlikum kui vastava eridialoogi kasutamine ning seetõttu kohustatakse kasutajat tehingule alati lisama selgitust. See võimaldab hiljem tehingu sisestamisel tehtud vigu parandada, sest tehinguobjekti küljes on kasutaja kirjeldus sellest, mida ta üritas kujutada.

Kasutaja eest sünkroniseeritakse tehingu kreediti ja deebeti vara summa lahtrite sisu, sest need summad on tavaliselt võrdsed. Summa lahtrite sisu ei sünkroniseerita juhul kui kreeditpoolel on toode või kui üks varadest on tühi hulk. Kui tehingu kreeditpoolel on toode, siis on tavaliselt tegu ostu-müügitehinguga, kus kreedit ja deebet ei pruugi olla võrdsed.

4.2.4 Inventuuride vaade

Inventuuride vaates kuvatakse sooritatud inventuure tabeli ridadena. Iga inventuuri kohta kuvatakse inventuuri sooritaja märkuseid, inventuuri sooritamise hetke ning ülelugemise tulemusi toodetele, taarale ja sularahale. Üle loetud varade taustaks olev ristkülik on värvitud kas roheliseks, kollaseks või punaseks vastavalt sellele, kas üle loetud kogus klappis eeldatava kogusega, vara oli eeldatust rohkem või vara oli eeldatust vähem. Juhul kui kogused ei klappinud, lisatakse varale *tooltip*, kus on kirjas, kui palju vastavat vara pidanuks olema. Näidet loetud koguste kuvamisest näeb jooniselt 12. Lisaks on igal inventuurireal nupud inventuuri muutmiseks ja kustutamiseks. Samamoodi nagu tehingute kustutamisel, küsitakse ka inventuuri kustutamisel kasutajalt kinnitust, sest tegu on pöördumatu protsessiga ning inventuuri tulemusi pole reaalelust võimalik taastada.



Joonis 12: Inventuuril üle loetud varade kogused. Juhul kui kogused ei klappinud, on varale lisatud *tooltip*, mis näitab kui palju eeldatavasti vara pidanuks olema.

Inventuure saab muuta ja luua dialoogiaknas. Uue inventuuri alustamisel palutakse kasutajal kokku lugeda sularaha, tühi taara ja laos olevate toodete kogused. Ülevaatlikkuse lisamiseks kuvatakse kasutajale kokkuvõtet inventuuri tulemustest, mida arvutatakse jooksvalt ümber iga kord, kui kasutaja mõnda sisendvälja muudab. Samuti juhitakse kasutaja tähelepanu sisendväljadele, mis on veel täitmata või kus ülelugemise tulemus ei klapi oodatud tulemusega. Nähes seda, saab joogivanem vastava vara koguse üle lugeda ning võib avastada, et on teinud lugemisel vea. Küll ei aita selline lähenemine olukordades, kus joogivanem on vara kiiruga valesti kokku lugenud ning saanud tulemuse, mis klapib eeldatava vara kogusega.

5 Edasiarendus

Töö käigus arendati välja *front-end*'i prototüüp joogivanema infosüsteemile ning *back-end*'i liidese spetsifikatsioon ja funktsionaalsed nõuded *back-end*'ile. Enne süsteemi kasutusele võttu tuleb *back-end* välja arendada ning testida *front-end*'i ja *back-end*'i koos. Hetkel on rakenduse *front-end*'i testitud vaid juhtudel, kus rakendus teeb serverile GET päringuid. Seda, mis juhtub pärast PUT, POST ja DELETE päringute saatmist *back-end*'ile, pole põhjalikult testitud.

Lisaks tuleb välja töötada lahendus kasutajate autentimiseks ja autoriseerimiseks. Antud töös on kirjeldatud kasutajarolle, kuid pole öeldud kuidas vastavaid piiranguid kasutajatele kehtestada. *Single-page application*'i puhul on üks võimalus autentimiseks kasutada *JSON Web Token* (JWT) lahendust, kus kõigi *back-end*'ile saadetavate HTTP päringute päisesse lisatakse n-ö allkirjastatud *token*, kus on kirjas kasutaja roll ning õiguste aegumise hetk. Klient saab *token*'i *back-end*'ilt sisse logides. Klient saab oma *token*'it küll muuta, kuid kui ta ei tea salasõna, millega *token* allkirjastatud on, siis saab server aru, et tegu ei ole valiidse *token*'iga ning ei luba kliendile ligipääsu andmetele. JWT tehnoloogia võimaldab autoriseeritust tõestada mitmele erinevale *back-end*'ile sama *token*'iga. See võimaldab tulevikus organisatsioonile välja arendada uusi *back-end*'e nii, et kasutaja ei pea end igaühe juures eraldi sisse logima.

Front-end'i tuleb lisada sisse logimise dialoog. Samuti tuleb muuta menüüriba, et näidata rakenduse kasutajale, kellena ta sisse logitud on ning mis on tema roll. Välja tuleb töötada vajalikud vaated seoses ameti üle andmisega ja järgmise semestri ametikandjate defineerimisega. Revisjonikomisjoni liikmetele tuleb võimaldada kontrollitava semestri tehingud külmutada pärast ameti üle andmist.

Vaatleme võimalusi, kuidas saaks rakendust veel edasi arendada siis, kui eelmainitud miinimum süsteemi kasutuselevõtuks on implementeeritud ja testitud.

1. Taara hinna muutmise võimaldamine. Taara hinda on Eesti Vabariigis varem muudetud ning miski ei välista, et see ei võiks tulevikus uuesti muutuda.

2. Pangakonto väljavõtte import või regulaarne pangakonto väljavõtte kontrollimine. Iga pangatehinguga on seotud arhiveerimistunnus ning selle abil saaks viia vastavusse kõik joogivanema infosüsteemis olevad tehingud pangakonto väljavõtte tehingutega. Joogivanema infosüsteem võiks analüüsida tehingute selgitusi ja osapooli ning lisada joogivanema infosüsteemi oma parima pakkumise selle kohta, milline tehing tuleks lisada. Kuna see tegevus on veaohtlik, tuleks lisada joogivanemale märkus tehingute kohta, mis on automaatselt konto väljavõttelt lisatud.

3. Tšeki pildistamine ja selle põhjal joogivanema infosüsteemi tehingu lisamine. Seda poleks vaja, kui oleks võimalik realiseerida pangakonto väljavõtte regulaarne automaatne importimine.

4. Triipkoodi lugemine toote tuvastamiseks. Joogivanem hakkab infosüsteemi müügitehinguid tihti sisestama nutitelefoniga. Kuna mobiilidel on kaamera, saaks seda kasutada kaubaartikli tuvastamiseks.

5. Kaubaartikli või liikme tuvastamine häältuvastusega. Eesti keele kõnetuvastusteekidega on võimalik saata helifail serverile ning vastu saada tekst. Juhul kui võimalikud vasted oleks võimalik piirata vaid joogivanema infosüsteemis leiduvate toodete ja

liikmete nimedega, siis on ehk võimalik häältuvastuse veaprotsent viia piisavalt madalale, et toote või liikme nime sisse trükkimise asemel on mugavam see verbaalselt öelda.

6. Automaatmeilid võlgu olevatele liikmetele tuletamaks neile meelde oma võlgnevuse suurust. Meilide saatmist peaks olema võimalik tähtajaliselt keelata juhuks, kui liikmega jõutakse kokkuleppele maksetähtaja osas.

7. Statistika vaade, kus näidatakse joogikassa varade koguste muutumist läbi aja ning muid huvipakkuvaid graafikuid.

8. Süsteem võiks püüda ennustada, millal mingi kaubaartikkel otsa lõppeb vastavalt tarbimise kiirusele ja laoseisule. Lisaks sellele, võiks süsteem koostada ostunimekirja joogivanemale, võttes arvesse seda, mis kaubaartiklite vastu on huvi ja planeerides koguseid nii, et erinevad kaubaartiklid lõppevad otsa enam-vähem samal ajal. See tähendab, et joogivanem ei pea nii tihti kaupa juurde ostma.

9. Serveripoolsed teated kliendile. Võimaldades serveril kliendile infot saata, saaks inventuuri sisestamise võimaldada mitmest seadmest korraga. Selleks peaks *back-end* ühest seadmest sisestatud info saatma kõigile teistele ühendatud klientidele.

Need arendused ei ole hädavajalikud, kuid pakuvad parajat väljakutset ning võimalust õppida. Siit on võimalik tulevastel informaatikatudengitest seltsi liikmetel projektiideid ammutada.

Kokkuvõte

Töö autor on EÜS Põhjalas joogivanema ametit pidanud kolmel semestril. See andis töö autorile hea arusaamise valdkonnast ning võimaldas näha joogivanema institutsiooniga seotud probleeme. Need teadmised mängisid olulist rolli kaalumisel, milline rakendus probleemid hästi lahendaks. Ameti pidamisega kaasnevate tüütute ja korduvate ülesannete täitmine oli tugev ajend infosüsteemi välja arendamiseks.

Töö tulemusena valmis kasutajaliidese prototüüp, nõuded *back-end*'ile ja rakendusliidese spetsifikatsioon. Infosüsteemi kasutusele võtmiseks vajalikud arendused planeeritakse valmis saada 2016/2017. õppeaasta sügissemestri alguseks. Uuele süsteemile üleminekul kaasnevate võimalike probleemide tuvastamiseks ja info kaotsimineku vältimiseks kasutatakse esimesel semestril uut ja vana süsteemi paralleelselt.

Kui süsteem rakendub EÜS Põhjalas edukalt, on põhimõtteliselt võimalik seda kasutada ka teistes üliõpilasorganisatsioonides. Kas antud infosüsteem teistele organisatsioonidele sobib, sõltub sellest, kui sarnased on joogivanema ametikoha ekvivalendi ülesanded vastavas organisatsioonis.

Infosüsteemi prototüübi välja arendamise käigus õpitu ning selle kohene rakendamine andis töö autorile kogemusliku arusaama sellest, kuidas klient ja server omavahel suhtlevad ning andis aimu sellest, kuhu veebirakenduste valdkond arenemas on. Töö autor peab vajalikuks välja tuua praktilise kogemuse väärtust asjade õppimisel. Mida rohkem on võimalik õpitut rakendada, seda sügavam mõistmine omandatakse õpitavast.

Viited

[1] Mike Cohn. *User stories applied: For agile software development*. Addison-Wesley Professional, 2004.

[2] Roy Thomas Fielding. *Architectural styles and the design of network-based software architectures*. PhD thesis, University of California, Irvine, 2000.

[3] Todd Fredrich. Restful service best practices: Recommendations for creating web services. `https://github.com/tfredrich/RestApiTutorial.com/raw/master/media/RESTful%20Best%20Practices-v1_2.pdf`, 2013.

[4] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7):1645–1660, 2013.

[5] Madhuri A. Jadhav, Balkrishna R. Sawant, and Anushree Deshmukh. Single page application using angularjs. *International Journal of Computer Science and Information Technologies*, 6(3):2876–2879, 2015.

[6] Nilesh Jain, Priyanka Mangal, and Deepak Mehta. Angularjs: A modern mvc framework in javascript. *Journal of Global Research in Computer Science*, 5(12):17–23, 2015.

[7] Javascript frameworks: The best 10 for modern web apps. `http://noeticforce.com/best-Javascript-frameworks-for-single-page-modern-web-applications`.

[8] e-teatmik: It ja sidetehnika seletav sõnaraamat. `http://www.vallaste.ee/index.asp`.

[9] Jim Webber, Savas Parastatidis, and Ian Robinson. *REST in practice: Hypermedia and systems architecture*. O'Reilly Media, Inc., 2010.

Kõiki internetiallikaid külastati viimati 08.05.2016.

Lisad

Lisa A *Front-end*'i prototüübi lähtekood

Prototüübi lähtekood on avalikult kättesaadav aadressil `https://github.com/madislutter/jis-klient`.

Lisa B *Back-end*'i *mock*'i lähtekood

Mock'i lähtekood on avalikult kättesaadav aadressil `https://github.com/madislutter/jis-server`.

Lisa C *Back-end*'i REST API liidese dokumentatsioon

Veebisaidi kujul dokumentatsioon on avalikult kättesaadav aadressil `http://www.pohjala.ee/~madis/jis/docs3.html`.

Lisa D Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina, **Madis Martin Lutter**,

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose
Joogivanema infosüsteemi *front-end*'i prototüüp üliõpilasseltsile,
mille juhendaja on Anne Villems,

1.1 reprodutseerimiseks säilitamise ja üldsusele kättesaadavaks tegemise eesmärgil, sealhulgas digitaalarhiivi DSpace lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

1.2 üldsusele kättesaadavaks tegemiseks Tartu Ülikooli veebikeskkonna kaudu, sealhulgas digitaalarhiivi DSpace'i kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.

2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.

3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tartus, **12.05.2016**