TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Liisi Lota Pals 185151IADB

TalTechi tudengiorganisatsioonide kohtumiste planeerimise rakendus PlanMeet

Bakalaureusetöö

Juhendaja: Meelis Antoi
Magistrikraad

Tallinn 2021

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: Liisi Lota Pals

17.05.2021

Annotatsioon

Käesoleva bakalaureusetöö eesmärk on luua süsteem, kus TalTechi tudengiorganisatsioonidel on võimalik oma kohtumisi planeerida arvestades ruumide ja inimeste vabu aegu. Lahendus vähendab kohtumiste planeerimiseks kasutatavate rakenduste arvu tuues kõik vajaliku ühte kohta ning tehes kogu protsessi tudengiorganisatsioonidele mugavamaks.

Arendusprotsessi käigus luuakse veebirakendus, mis sisaldab kalendrivaadet koos kohtumiste lisamise, muutmise ja kustutamisega, ühiste aegade leidmiseks küsitluste loomist, muutmist, kustutamist ja neile vastamist ning integratsiooni õppeinfosüsteemiga, et ruumide vabu aegu näha ja broneerida. Arendus on jagatud kahte osasse: serveripoolse ja kliendipoolse rakenduse arendus. Arendusprotsessi tulemuseks on töötav rakendus.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 65 leheküljel, 6 peatükki, 17 joonist.

Abstract

PlanMeet Application for TalTech Student Organizations to Plan Their Meetings

The aim of this bachelor's thesis is to create a system for TalTech student organizations to plan their meetings while bringing together students and rooms free times. Ready solution reduces the number of applications used to plan meetings by bringing everything needed to one place and making whole process more convenient for student organizations.

During the development process, a web application is created that includes calendar view together with adding, editing and deleting meetings, creating, editing, deleting and answering to questionnaires to find common times and integration with student information system to view and book rooms. The development process is dividend into two parts: server-side and client-side application development. The result of the development process is a running application.

The thesis is in Estonian and contains 65 pages of text, 6 chapters, 17 figures.

Lühendite ja mõistete sõnastik

API	Rakendusliides (*Application Programming Interface*)
CLI	Käsurea liides (*Command Line Interface*)
CSS	Märgistuskeel (*Cascading Style Sheets*)
DTO	Andmesaateobjekt (*Data Transfer Object*)
HTML	Märgistuskeel (*Hyper Text Markup Language*)
HTTP	Võrguprotokoll (*HyperText Transfer Protocol*)
IT	Infotehnoloogia
JDBC	Java andmebaasi ühenduvus (*Java Database Connectivity*)
JPA	Java püsivuse rakendusliides (*Java Persistence API*)
JSON	Andmevahetusformaat (*JavaScript Object Notation*)
JWT	JSON veebimärk (*JSON Web Token*)
LCD	Vedelkristallkuvar (*Liquid Crystal Display*)
MVC	Tarkvara arhitektuurimuster (*Model-View-Controller*)
ORM	Programmeerimistehnoloogia (*Object-relational Mapping*)
REST	Tarkvaraarhitektuuri laad (*Representational State Transfer*)
SQL	Päringukeel andmebaasiga suhtluseks (*Structured Query Language*)
TalTech	Tallinna Tehnikaülikool
TV	Televisioon
UI	Kasutajaliides (*User Interface*)
ÕIS	Õppeinfosüsteem

Sisukord

Jooniste loetelu

1 Sissejuhatus

Käesoleva töö eesmärk on luua süsteem, kus TalTechi tudengiorganisatsioonidel on võimalik oma kohtumisi planeerida arvestades ruumide ja inimeste vabu aegu. Arendusprotsessi käigus luuakse veebirakendus, mis sisaldab kalendrivaadet, küsitlusi ühiste aegade leidmiseks ja integratsiooni ÕIS-iga (õppeinfosüsteemiga), et ruumide vabu aegu näha ja saata taotlusi broneerimiseks.

Praegu kasutavad TalTechi tudengiorganisatsioonid oma kohtumiste planeerimiseks mitmeid erinevaid lahendusi. Ühiste aegade leidmiseks kasutatakse ühte, ruumide vabade aegade nägemiseks ja broneerimiseks teist või isegi rohkemaid erinevaid rakendusi. Ruumide ja inimeste vabade aegade leidmiseks puudub hetkel ühine lahendus. TalTechis on väga palju erinevaid tudengiorganisatsioone ja kohtumiste planeerimise lihtsustamine säästaks nende aega ning võimaldaks veel rohkem ja suuremaid asju korda saata. Selline süsteem lihtsustaks ka organisatsioonide vahelist koostööd ja ühiste ürituste korraldamist.

Hetkel kasutatakse ühiste aegade leidmiseks peamiselt veebirakendust Doodle. Kasutatavateks ruumideks on peamiselt TalTechi ruumid ja Mektory ning nende broneerimiseks saadetakse e-maile või kasutatakse ÕIS-i.

Lahendusena arendatakse bakalaureusetöö raames veebirakendus, mille abil on võimalik küsitluste kaudu leida inimestele sobivad ajad ja integratsioonist ÕIS-iga ruumide vabad ajad ning valida seeläbi sobivad ajad ja kohad oma kohtumiste korraldamiseks. Tudengiorganisatsioonide tegevus muutub seeläbi lihtsamaks ning annab neile võimaluse rohkem ettevõtmisi korda saata nii organisatsiooni sees kui ka erinevate organisatsioonide vahel.

1.1 Metoodika

Antud lõputöö raames kirjeldatakse esmalt lõputöös lahendatav probleem, analüüsitakse olemasolevate lahenduste häid ja halbu külgi ning tulemusena pakutakse välja IT lahendus, mis vastaks püstitatud eesmärkidele.

Järgnevalt pannakse paika erinevate kasutajagruppide vajadused ja seeläbi rakenduse funktsionaalsed nõuded. Rakenduse arendamiseks valitakse sobivaimad tehnoloogiad võrreldes erinevaid võimalikke kaasaegseid tehnoloogiad veebirakenduse arendamiseks. Kasutajagruppide vajadustel põhinedes luuakse rakenduse arhitektuur, andmebaasi mudel ning kasutajakogemuse ja kasutajaliidese disain.

Rakenduse arenduse protsess on kirjeldatud nii serveri- kui ka kasutajaliidese poolt. Lõpus hinnatakse valminud rakenduse kvaliteeti ja funktsionaalsust ning pakutakse välja võimalikke rakenduse edasiarendusi tulevikuks.

2 Probleemi ülevaade

Tallinna Tehnikaülikoolis on TalTechi kodulehe andmetel koos 12 akadeemilise korporatsiooni ja 10 kultuurikollektiiviga kokku 41 tudengiorganisatsiooni [1]. Lisaks on TalTechis veel viis üliõpilaskogu ja üliõpilasesindus. Tudengielul on oluline osa iga tudengi ülikooli kogemusest ja tudengiorganisatsioonide edukas toimimine on osa ülikooli edukast toimimisest.

TalTechi tudengiorganisatsioonidel on mitmeid erinevaid tegevusalasid, kuid kohtumisi korraldavad kõik. On need siis koosolekud, proovid, trennid, suuremad või väiksemad üritused, aga koos toimetamine ongi üks tudengiorganisatsiooni peamisi alustalasid. Kohtumiste üks keerukamaid osasid on aga ühiste aegade leidmine. Praegu kasutatakse selleks peamiselt veebirakendust Doodle.

Teine pool kokkusaamiste planeerimisest on asukoht. Praeguse koroonakriisi ajal korraldatakse kohtumisi tihti ka veebi kaudu, kuid kui siiski füüsiliselt kokku saada, siis peab eriti läbi mõtlema, et ruum oleks osalejate arvule vastav. Lisaks on tudengiorganisatsioonidele oluline ka ruumi asukoht. Asukoht võiks näiteks olla organisatsiooni peakontori lähedal, et sealt oleks lihtsasti võimalik tehnikat ja muud vajalikku tuua. Proovide ja trennide puhul võivad olulised olla näiteks ruumi kõla ja vaba põrandapinna olemus. Mõnel juhul võib oluline olla ka näiteks vajaliku tehnika olemasolu.

Peamisteks kasutatavateks ruumideks on TalTechi ruumid ja Mektory. TalTechi ruumide vabu aegu broneeritakse eelkõige läbi ÕIS-i, aga seda tehakse ka e-maili teel. Üliõpilasesindusel on ka koosolekuruumid, mille vabu aegu näeb nende Google Calendar-ist, aga broneerimiseks tuleb e-maile saata. TalTechi tudengimaja vabu aegu näeb Outlook kalendrist ja broneerimine käib e-maili teel. Mektory vabu aegu on võimalik vaadata eSpotIt lehel, kus on võimallik ka broneerida, kuid seal on ruumide kasutamine tasuline. TalTechi tudengiorganisatsioonidel on Mektory ruume võimalik kasutada eritingimustel ja neid broneeritakse üldiselt e-maili teel. TalTechi ruumidest on

erilised veel ka näiteks saunad, mille kasutamine on tasuline ja selleks sõlmitakse ka leping. Broneerimine käib e-maili teel.

Erinevate eksisteerivate kohtumiste planeerimise lahenduste probleemiks on integratsiooni puudumine ÕIS-iga. ÕIS-is on probleemiks inimeste vabade aegade leidmise võimaluse puudumine ja ruumide leidmise ebamugavus. Olemasolevatest lahendustest on võimalik analüüsida rakendusi inimeste ühiste aegade leidmiseks ning rakendusi ruumide broneerimiseks. Analüüsi tulemusena on võimalik kirja panna rakenduse skoop.

2.1 Ühiste aegade leidmise eksisteerivad lahendused

Ühiste aegade leidmise rakenduste analüüsimiseks valiti Doodle ja kalendrirakendused Microsoft Outlook Calendar ja Google Calendar. Doodle on peamine rakendus, mida praegu tudengiorganisatsioonid ühiste aegade leidmiseks kasutavad ja kalendrirakendustena analüüsitakse kahte kõige populaarsemat [2].

2.1.1 Doodle

Doodle on Šveitsi veebirakendus, mis võimaldab leida meeskonnale sobivaima aja koosolekuks. Seal saab luua veebiküsitlusi võimalike aegadega ja vastajad saavad märkida neile sobivad ajad. Doodle on tegutsenud aastast 2007. Rakendust on võimalik siduda ka erinevate kalendri rakendustega nagu Google Calendar ja Outlook Calendar, kuhu on võimalik automaatselt lõplik koosoleku aeg lisada [3].

Küsitlust luues on võimalik valida erinevaid kuupäevi ja kellaaegu. Valida on võimalik ka terved päevad. Seadistada on võimalik ka seda, kas vastus võib olla ainult jah või ei või ka võib-olla, kui palju inimesi võib osaleda ühel ajal, kas valida võib ühe või rohkem aegu ja kas vastused on kõigile nähtavad või ainult küsitluse loojale. Tasulises versioonis on võimalik määrata ka küsitlusele vastamise tähtaeg ja meeldetuletuste saatmine. Vastajad sisestavad oma nime ja valivad neile sobivad ajad. Küsitluse looja saab lõpus valida sobivaima aja, sellega küsitluse sulgeda ja ürituse kalendrisse lisada.

Osalejate teavitamiseks e-maili teel pakub rakendus võimalust teha seda manuaalselt e-maile sisestades, kuid kuna kontaktandmeid tasuta versioonis ei koguta, siis automaatselt seda teha ei saa. Küsitluses on võimalik lisada ka asukoht, kuid võimalus

ruumidega integreerimiseks puudub. Samas on rakendusel integratsioon Zoom-i ja Microsoft Teams-iga, mis tähendab, et on võimalik virtuaalsete kohtumiste linke automaatselt lisada oma üritustele. Väga kasulikud on ka tasulise versiooni omadused nagu vastamise tähtaja lisamine, meeldetuletused ja automaatne lõpliku kohtumise aja edastamine, kuid tudengiorganisatsioonidel üldiselt pole suurt sissetulekut ja tasuliste versioonide kasutamine on variant vaid äärmisel vajadusel. Tasuta versioonis saab ürituse otse kalendrisse lisada ka vaid korraldaja ja mitte vastajad.

2.1.2 Microsoft Outlook Calendar ja Google Calendar

Nii Microsoftil kui ka Google-il on oma kalendrirakendused, mis on ühed populaarseimad [2]. Need on osa suurematest pakettidest, mis hõlmavad näiteks e-maili funktsionaalsust. Esimene Microsoft Outlook Calendar versioon pärineb aastast 1998 [4]. Google Calendar loodi aastal 2006, aga avaldati aastal 2009 [5]. Outlook on osa Microsoft Office-i tellimusest ning on kasutuses ka TalTechis nii õpetajatel kui õpilastel. Samas ei tähenda see, et see kõigil TalTechi tudengiorganisatsioonide liikmetel kasutuses oleks, sest kõik liikmed pole TalTechi tudengid.

Kalendrirakenduste peamiseks funktsionaalsuseks on aja planeerimine. Seal saab järge pidada oma personaalsete tegemiste üle, aga ka ühiseid üritusi korraldada. Mõlemas rakenduses on võimalik luua üritusi ja teistele kutseid saata. Kutseid on võimalik saata ka Outlooki ja Google-i vahel, kuna toetatakse sama failitüüpi iCalendar [6]. Outlookis on ka selline funktsionaalsus, kus näeb juba üritust luues, kas kutsutaval inimesel on see aeg vaba või mitte. Samas toimib see vaid Outlooki kasutajate vahel ja inimestel võib olla ka kalendriväliseid kohtumisi, mis tähendab, et lisakinnitus on ikka vajalik. Kasulik on integratsioon emailiga, mis võimaldab üritusel osalemist ühe nupuvajutusega kinnitada või tagasi lükata otse e-mailist. Üritust luues on võimalik seadistada mitmeid erinevaid parameetreid nagu alguse aeg, lõpu aeg, asukoht ja kirjeldus. Üritust on võimalik ka korrata näiteks iga nädal või mingi muu tihedusega. Mõlemal kalendril on võimalik valida ürituste jaoks ka erinevaid värve ja luua mitmeid erinevaid kalendreid, et eristada näiteks personaalseid ja tööüritusi. [7] [8]

Peamiseks probleemideks kalendrirakenduste puhul on integratsiooni puudumine ÕIS-iga. Seda oleks võimalik teha, kuid selleks tuleks muuta kogu olemasolevat ruumide broneerimise süsteemi TalTechis.

2.2 Ruumide broneerimise eksisteerivad lahendused

Ruumide broneerimise rakenduste analüüsimiseks valiti eelkõige ÕIS, kus praegu peamiselt TalTechi ruume broneeritakse. Lisaks analüüsitakse kolme muud rakendust ruumide broneerimiseks: Plandok, Calendly ja Microsoft Bookings. Need on heaks eeskujuks, kuidas loodavas rakenduses ruumide broneerimine võiks toimida, kuid neid saaks TalTechi ruumide broneerimiseks kasutada vaid siis, kui kogu ruumide haldus sinna üle viidaks.

2.2.1 ÕIS

Ruumide broneerimiseks TalTechis kasutatakse praegu peamiselt ÕIS-i. Kasutajaliides on aga ebaintuitiivne ja selle mõistmine võtab aega. Ruumi otsimisel on võimalik valida, kas „Genereeri ajad“ või „Lisa oma aeg“. Juba see valik tekitab natuke segadust nagu annaks see võimaluse ise ruumidele vabu aegu lisada. Tegelikult tähendavad need aga võimalusi määrata korduvat või ühekordset aega. Ühekordse broneeringu tegemiseks tuleb sisestada kuupäev ja alguse ning lõpu kellaaeg. Korduva broneeringu jaoks tuleb määrata semester, alguse ja lõpu kellaaeg, nädalapäev ja mis nädalatel semestrist.

Edasi saab määrata parameetreid, mille järgi ruumi otsida nagu ruumi number, hoone, korrus, kohtade arvu miinimum ja maksimum, kas ruum on avalik või mitte, kas ruum peab olema vaba ja varustuse. Varustusest on valikus arvutiga töökoht, dataprojektor, dokumendikaamera, helivõimendus, kaldauditoorium, kriiditahvel, LCD ekraan/TV, mikrofon, mööbli liigutamise võimalus, reguleeritav valgustus, tahvel ja videosalvestus. Tulemuseks saadakse list parameetritele vastavatest ruumidest.

Probleemiks on lisaks segadust tekitavale kasutajaliidesele ka näiteks see, et pole võimalik valida mitut asukohta – ruumi, hoonet, korrust. Mitmes erinevas hoones ruumide otsimiseks tuleb praegu mitu otsingut teha. Lisaks on võimalik otsida vaid kindlat aega, aga kui sobivad mitmed erinevad ajad, siis tuleb samuti mitu otsingut teha.

Kui sobiv ruum on leitud, siis tuleb broneerimiseks valida broneerimise tüüp: konsultatsioon, eksam, õppe- või teadustöö või muu. Lisada saab ka täpsustava kommentaari ja valida, kelle nimele ruum broneeritakse.

2.2.2 Calendly

Calendly on aja planeerimise tarkvara. See on integreeritav Google Calendar-i ja Outlookiga. Selle abil on võimalik saata oma kalendri link teistele ning teised saavad broneerida kohtumiseks aja sinu kalendris. Broneeritud kohtumised saab automaatselt lisada Google-i või Outlooki kalendrisse [9] [10]. Süsteemi on võimalik kasutada nii inimeste aja kui ka ruumide broneerimiseks. TalTechi ruumide selliseks broneerimiseks oleks jällegi vaja kogu olemasolevat ruumide halduse süsteemi muuta. Inimeste aegade broneerimisel ei piisa alati jälle kalendri nägemisest. Kõiki üritusi ei panda kalendrisse ja üle küsimine on igal juhul vajalik. Eriti just tudengiorganisatsioonide puhul, kus pole kindlat tööaega ja kohtumised toimuvad nii päeval kui ka õhtul, nädala sees ja nädalavahetusel.

2.2.3 Plandok

Plandok on rakendus kohtumiste planeerimiseks. See sobib nii ettevõtte siseselt töötajate ajakavade haldamiseks kui ka klientidele kohtumiste broneerimiseks. Seal on võimalik lihtsasti nii uusi broneeringuid teha kui ka tühistada. Kõigi kohtumiste kohta saadetakse klientidele ka meeldetuletused, mille sisu on võimalik endale sobivaks seadistada. Plandok salvestab ka klientide kohtumiste ajaloo ja eelistused ning võimaldab seeläbi paremat teenust pakkuda. TalTechis kasutusele võtmiseks oleks taaskord vaja olemasolevat ruumide halduse süsteemi muuta [11].

2.2.4 Microsoft Bookings

Microsoft Bookings on rakendus kohtumiste broneerimiseks. See koosneb broneerimise kalendrist klientidele ja integratsioonist Outlook Calendar-iga, mille abil jõuavad kohtumised töötajate kalendritesse. Kohtumisi saab korraldada ka Microsoft Teams-is või Skype-is. Teams-i jaoks on olemas ka rakendus ning kogu planeerimist on võimalik teha Teams-i sees. Lisaks on olemas kinnituskirjade saatmine klientidele koos kalendrikutse või Teams-i kohtumise lingiga. Microsoft Bookings on osa Microsoft 365 pakettidest [12] [13]. TalTechi tudengitele on võimaldatud Microsoft 365 kasutamine, aga tudengiorganisatsioonides ei pruugi olla ainult TalTechi tudengid. TalTechis kasutusele võtmiseks oleks taaskord vaja olemasolevat ruumide halduse süsteemi muuta.

2.3 PlanMeet rakenduse skoop

Lahendust, mis võimaldaks samal ajal inimeste ühiseid aegu ja TelTechi ruumide vabu aegu vaadata hetkel ei eksisteeri. Ühiste aegade leidmiseks kasutatakse praegu peamiselt Doodle-t. Ruume broneeritakse peamiselt ÕIS-is. ÕIS-is ruumide broneerimise kasutajaliides on hetkel ebamugav ning puudub võimalus otsida mitut hoonet, korrust või ruumi korraga.

Üks lahendus oleks viia kogu ruumide broneerimise haldus üle mõnele teisele platvormile nagu Calendly, Plandok või Microsoft Bookings. Seal oleks võimalik iga ruumi jaoks eraldi kalender tekitada ning neid saaks mugavalt broneerida. Samas kasutavad ÕIS-i ka õppejõud ja TalTechi töötajad ning inimeste ümber koolitamine võtaks palju aega. ÕIS-iga on juba harjutud ning paljud ei pruugi näha praeguse lahenduse probleeme. Eeltoodud ruumide broneerimise rakendustesse ÕIS-i integreerimine oleks aga väga keerukas või isegi võimatu.

Praeguse broneerimise süsteemi säilitamise ja tudengiorganisatsioonidele mugavama broneerimise jaoks pakub töö autor välja lahenduse luua kohtumiste planeerimise rakendus PlanMeet. Rakenduse üheks osaks on kalendrivaade, kus näeb erinevaid üritusi – väiksemate gruppide vahel, organisatsioonisiseseid ja organisatsioonide vahelisi. Üritusi saab filtreerida ning ürituse loomisel on võimalik valida, kes seda näevad – kas kindlad inimesed, kindlad organisatsioonid või kõik organisatsioonid. Kalendrisse on võimalik lisada ka tähtpäevi nagu näiteks organisatsiooni sünnipäev, organisatsiooni liikmete sünnipäevad ja üliõpilaskonna sünnipäev. Rakenduse teiseks osaks on küsitlused, et selgitada välja millal liikmed saaksid koosviibimistel osaleda. Küsitluste kõrval saab otsida sobivaid vabu TalTechi ruume ja saata taotlusi nende broneerimiseks. Ruumide info ja broneerimine käib läbi ÕIS-i integratsiooni.

Ruumide otsimiseks on võimalik määrata erinevaid parameetreid. Kohustuslikuks parameetriks ÕIS-is on aeg. Valida tuleb, kas ühekordne üritus või korduv ning ühekordse aja puhul kuupäev ning alguse ja lõpu kellaaeg ja korduva ürituse puhul semester, alguse ja lõpu kellaaeg ning nädalapäev. Korduva ürituse puhul on võimalik täpsustada lisaks otsitavad nädalad. Otsinguparameetritena on võimalik lisada veel ruumi number, hoone ja korrus. ÕIS-is on hetkel neid kõiki võimalik valida vaid üks, kuid pakutavas lahenduses oleks võimalik korraga valida mitu. Lisaks on võimalik määrata kohtade arvu miinimum

ja/või maksimum ning varustus. Varustusest on valikus arvutiga töökoht, dataprojektor, dokumendikaamera, helivõimendus, kaldauditoorium, kriiditahvel, LCD ekraan/TV, mikrofon, mööbli liigutamise võimalus, reguleeritav valgustus, tahvel ja videosalvestus.

Kui sobivad aeg ja ruum on leitud, siis tuleb ruum broneerida. Selleks on vaja määrata, kelle nimele TalTechi tudengitest ruum broneeritakse. Broneerimise tüübiks valitakse automaatselt „Muu“ ning kirjeldusse genereeritakse lause selle kohta, et taotlus tuli rakenduse PlanMeet kaudu ning lisatakse ka organisatsiooni ning ürituse nimed. Järgmisena saadetakse taotlus ÕIS-i. Kui ühine aeg ja asukoht on leitud, siis on võimalik küsitlus sulgeda ja kohtumise juures vastavad väljad täita. Kohtumine lisatakse PlanMeet kalendrisse ja osalejatele saadetakse meilile link ürituse oma Google-i või Outlooki kalendrisse lisamiseks.

3 PlanMeet rakenduse analüüs

3.1 Rakenduse nõuded

Kirjeldatud rakenduse nõuded põhinevad antud lõputöö skoobil. Nõuete määramisel on arvesse võetud eelkõige töö autori enda kogemusi tudengiorganisatsiooni liikmena.

Nõuded on kirjeldatud kasutajalugudega ning grupeeritud lugudesse. Kasutajalugude rollideks on tudengiorganisatsiooni liige, kohtumise korraldaja, kohtumisel osaleja ja administraator. Kohtumiste korraldajaks ja osalejaks võivad olla kõik tudengiorganisatsioonide liikmed, kelle kuuluvus organisatsiooni on kinnitatud administraatori poolt. Administraatoriks on isik, kellel on ligipääs organisatsioonide registrisse ning kes saab uusi kasutajaid rakendusse lisada. Kasutajalood kirjeldavad nii nõudeid funktsionaalsusele kui ka visuaalsele kujundusele.

Nõuded on välja toodud lisas 2.

3.2 Serveripoolse rakenduse tehnoloogiad

Serveripoolse rakenduse arendamiseks tuleb valida eelkõige programmeerimiskeel. Arendamise lihtsustamiseks on tänapäeval võimalik kasutada ka mitmeid erinevaid raamistikke, mis vähendavad baasfunktsionaalsuse jaoks vajalikke koodiridu sisaldades valmis komponente. Lisaks tuleb plaanitava rakenduse andmete hoiustamiseks valida andmebaas.

3.2.1 Programmeerimiskeele valik

Serveripoolseks programmeerimiskeeleks sobivad veebirakenduse arendamisel mitmed erinevad. Lõputöö raames tasub kasutada mõnda õpingutel käsitletutest, et olemasolevaid teadmisi proovile panna ja laiendada. Arvestada tasub kindlasti ka kaasaegseid tavasid veebirakenduste arendamiseks, et kogemus tuleks kasuks tulevikus erialasele tööle asudes. Võimalikud variandid:

- Java – objektorienteeritud programmeerimiskeel, pikka aega väga populaarne olnud, stabiilne, laialt kasutuses nii veebirakenduste kui ka mobiilirakenduste arendamisel [14] [15]. Autoril on kogemust antud keelega praktikal käimisest ja koolis käsitleti seda samuti veebirakenduse arendamiseks piisaval tasemel, seega plaanitava rakenduse arendus ei vaja palju juurde õppimist.

- C# - objektorienteeritud programmeerimiskeel, arendatud Microsofti poolt, kasutatakse laialt serveripoolseks programmeerimiseks, mängude ja veebirakenduste arendamiseks [15] [16]. Autoril on kogemust antud keelega koolist, kus käsitleti seda üsna põhjalikult ning arendati veebirakendus. Plaanitava rakenduse arendamiseks on tase piisav ega vaja palju juurde õppimist.

- PHP – objektorienteeritud programmeerimiskeel, võimalik lihtsasti HTML-iga siduda, puuduvad ranged reeglid ning suurtes projektides võivad muutuda raskesti loetavaks [16] [17]. Autori kogemus on algtasemel ning suure rakenduse ehitamine vajaks palju juurde õppimist. Lisaks on plaanitav rakendus küllaltki suuremahuline ning loetavus oluline.

- Python – objektorienteeritud programmeerimiskeel, sobib hästi algajale, sobib nii skriptikeeleks kui ka veebirakenduste arendamiseks, üks populaarsemaid tänu oma lihtsusele [16] [14] [18]. Autori kogemus selle keelega on teistega võrreldes mõnevõrra väiksem, sest koolis käsitleti esimesel semestril ning veidi algsemal tasemel. Suuri süsteeme ei ehitatud ja selle meelde tuletamine ning juurde õppimine oleks aeganõudev.

Eelnevaid punkte arvestades on mõistlik valida, kas Java või C#. Mõlemaga on autoril piisavalt kogemust ning mõlemad sobivad hästi veebirakenduse arendamiseks. Tööpakkumisi Eestis on rohkem Java arendajatele. Kuna töökogemus on autoril Java-ga ning sellega on plaanis töömaastikul ka jätkata, siis on hea lõputööd kasutada enda arendamiseks selles ning valida rakenduse serveripoolseks arendamiseks Java.

3.2.2 Raamistike valik

Raamistikud annavad rakendusele struktuuri ning lihtsustavad oluliselt arendust. Mitmeid tegevusi on võimalik nende abil automatiseerida. Valides serveripoolseks programmeerimiskeeleks Java, on veebirakenduse ehitamiseks üks kõige populaarsem

raamistik Spring. See võimaldab luua nõrgalt seotud mooduleid, mille sidumine toimub raamistiku poolt. Spring raamistik sisaldab ka mitmeid kasulikke valmis mooduleid nagu näiteks Spring JDBC, Spring MVC ja Spring Security, mis vähendavad rakenduse arendusaega [19] [20]. Arvestades rakenduse nõudeid ja autori kogemust Spring raamistikuga nii koolist kui ka praktikalt, annaks selle kasutamine plaanitavas rakenduses kõige parema lõpptulemuse.

Spring raamistikul on veel mitmeid kasulikke laiendusi nagu näiteks Spring Boot. Spring Boot-i abiga on võimalik lihtsustada rakenduse seadistamist ja ehitamist ning see sisaldab ka serverit rakenduse ehitamiseks. See vähendab sõltuvuste arvu, mida peab manuaalselt lisama ning lisab need lähtekoodi kompileerimise ajal automaatselt. Antud lõputöö raames loodav serveripoolne rakendus peab pakkuma REST API päringute funktsionaalsust ja Spring Boot sobib väga hästi selle saavutamiseks [21] [22].

Üks osa rakenduse arendamisest, mida on võimalik raamistike abil lihtsustada, on suhtlus andmebaasiga. ORM (*Object Relational Mapping*) on tehnoloogia, mis võimaldab siduda andmemudeli objekte relatsioonilise andmebaasiga. See võimaldab andmeid lihtsasti konverteerida, genereerib SQL päringud rakenduse kompileerimise ajal automaatselt ning võimaldab arendajal keskenduda rakenduse funktsionaalsuse loomisele ilma, et peaks iga päringu tegemisel kontrollima selle turvalisust ning SQL koodi korrektsust [23]. Teine võimalus andmebaasiga suhtluseks on JDBC. See lisab rakendusele abstraktsiooni kihi, mis võimaldab SQL päringuid sooritada. See avab ja sulgeb ühenduse, sooritab SQL päringud ja tegeleb erinditega. Samas kirjutatakse SQL päringud sel juhul otse koodi ning iga muutusega andmebaasis tuleb muuta ka koodi. ORM-i puhul seevastu on andmebaasi ja domeenid objektid algusest peale ühendatud ja SQL päringuid koodi kirjutama ei pea [24] [25].

ORM-i kasutamine plaanitavas rakenduses lihtsustaks oluliselt arendust, sest aitaks vältida SQL lausete koodi kirjutamist. Samuti väldib see probleeme andmebaasi ja domeeni objektide sidumisel. Üheks populaarseimaks ORM raamistikuks on Hibernate. Hibernate-ist on välja kasvanud ka standard JPA (*Java Persistance API*), mida üldiselt ORM-i puhul kasutatakse. Hibernate on üks populaarseim JPA implementatsioon, mida on sobilik kasutada ka plaanitavas rakenduses [24].

3.2.3 Projekti ehitamise tööriista valik

Projekti ehitamise tööriistade abil on võimalik projekti ehitada ja hallata väliseid sõltuvusi. Java projekti ehitamise tööriistadena kasutatakse peamiselt Gradle-t ja Maven-it. Maven on juba kaua kasutuses olnud, Gradle samas on uuem ja hoogsalt arenenud viimastel aastatel. Gradle eelisteks on lühem süntaks ja kiirus. Kiiruse annavad koodi muutuste jälgimine, varasemate ehituste informatsiooni säilitamine ja taaskasutus. Samas paljudes eksisteerivates rakendustes on kasutusel Maven. Kuna antud töös arendatakse täiesti uus rakendus, siis kasutatakse Gradle-t, mis on kiirem ja võimaldab vähem koodi kirjutada [26] [27] [28].

3.2.4 Andmebaasi valik

Andmebaase võib jaotada erinevatesse kategooriatesse: hierarhiline, võrgu, objektorienteeritud, relatsiooniline ja mitte relatsiooniline. Veebirakenduste jaoks kasutatakse peamiselt relatsioonilisi andmebaase, mis põhinevad relatsioonilisel andmemudelil. Andmed on paigutatud ridade ja veergudena tabelisse. Igal tabelil on oma võti, mis eristab seda teistest ja kõik tabelid on omavahel seotud. Andmed on paigutatud range skeemi järgi, mis tähendab, et tulemused on alati oodatavad ja kergesti töödeldavad. Andmete salvestamiseks, muutmiseks ja vaatamiseks kasutatakse SQL-i [29] [30]. Võimalikud populaarsed relatsioonilised andmebaasid: MySQL, PostgreSQL, Oracle, MariaDb, SQLite ja Microsoft SQL Server.

Oracle on väga sobilik suurtele ettevõtetele suurte andmemahtude hoidmiseks. Installerimine võib olla samas keerukas ja vajada lisaressursse. Microsoft SQL Server sobib samuti hästi suurtele ettevõtetele, kes kasutavad Microsoft tooteid [31]. Plaanitava rakenduse jaoks ei ole vaja nii võimsat andmebaasi ning üles seadmine ja haldus peaksid olema võimalikult lihtsad. Ka PostgreSQL on üks populaarne võimalik lahendus, kuid dokumentatsiooni on selle kohta vähem ning konfiguratsioon on keerukas [31].

MySQL on üks kõige populaarsemaid andmebaase veebirakenduste jaoks. Sellel on mitmeid erinevaid häid kasutajaliideseid haldamiseks. MySQL alternatiiviks on MariaDb, mis erinevalt MySQL-ist on tasuta ka ettevõtetele [31]. Veel üks võimalik lahendus on SQLite. See on palju lihtsam võrreldes teistega. See ei vaja konfiguratsiooni ega serverit. Andmed salvestatakse arvuti kõvakettale [32]. Selline lahendus poleks aga jätkusuutlik, kui kasutajate maht tulevikus suureneb.

Arvestades populaarsust, hinda ja kasutusmugavust valitakse plaanitava rakenduse jaoks MariaDb andmebaas, mis on tasuta, lihtsasti konfigureeritav ja hallatav ning samade võimalustega kui MySQL.

3.3 Kliendipoolse rakenduse tehnoloogiad

Kliendipoolse rakenduse arendamiseks tuleb valida programmeerimiskeel ja mõistlik on sinna kõrvale valida ka üks populaarseimatest raamistikest, mis lihtsustavad veebirakenduse arendust valmis komponentide kaudu.

3.3.1 Programmeerimiskeele valik

Peamisteks valikuteks kliendipoolse rakenduse arendamisel on JavaScript ja TypeScript. TypeScript on JavaScripti edasiarendus. Peamine erinevus seisneb tüüpides - JavaScripti puhul pole objektidel tüüpe, aga TypeScriptis on. Tüübid parandavad koodi arusaadavust. Samas on JavaScriptil suur aktiivne arendajate kogukond. Suuremate rakenduste puhul soovitatakse kasutada TypeScripti kui objektorienteeritud programmeerimiskeelt [33].

Plaanitav kliendipoolne rakendus on küllaltki suuremahuline, seega kasutusele võetakse TypeScript. Lisaks kasutatakse kliendipoolse rakenduse arendamiseks märgistuskeelt HTML (*Hyper Text Markup Language*), milles enamik veebilehti kirjutatud on [34]. Kujundamiseks kasutatakse veel lisaks CSS-i (*Cascading Style Sheets*), mis võimaldab HTML elementide fonti, värve ja paigutust muuta [35].

3.3.2 Raamistike valik

Raamistikud võimaldavad kasutada valmis kirjutatud komponente ning vähendada koodiridu ja arendusaega. Kolm kõige populaarsemat raamistikku kliendipoolse rakenduse ehitamiseks on Angular, React ja Vue [36].

React on Facebooki poolt arendatud raamistik, mille algsed eesmärgid olid jõudlus ja tõhus kasutajaliides. See avaldati aastal 2013 ja põhineb JavaScriptil. React võimaldab luua veebirakenduse kasutajaliidest koos dünaamiliste HTML komponentidega. Tavapäraseks on JavaScripti ja HTML-i hoidmine samas failis, et tagada vaate selgus. Reacti võib pidada üheks populaarseimaks raamistikuks, millel on suur globaalne kogukond ja rohkelt avatud lähtekoodiga komponente. Reacti probleemideks võib pidada

keerukust alustajale ja poolikut dokumentatsiooni, mis on tingitud arenemise kiirusest [36].

Vue on Evan You poolt arendatud raamistik aastast 2014, mis võimaldab lahendada dünaamilisi protsesse ilma rohkete raamistiketa. Vue põhineb JavaScriptil. See on väiksemahuline ning aitab seeläbi tõsta kasutajaliides jõudlust. Vue-l on sisukas ja konkreetne dokumentatsioon, mis lihtsustab selle õppimist algajal. Probleemiks võib pidada väikest kasutajate kogukonda [36].

Angular on Google-i poolt arendatud raamistik, mis põhineb TypeScriptil. See avaldati aastal 2016 ja järgib objektorienteeritud programmeerimise põhimõtteid. Seda peetakse üheks kõige terviklikumaks raamistikuks, sest paljusid probleeme on võimalik lahendada raamistiku siseselt ilma väliste teekideta. Probleemideks võib pidada keerukamat mitmekihilist arhitektuuri ja TypeScripti kasutamist JavaScripti asemel [36].

Antud töös kasutatakse raamistikku Vue, mis on väiksemahuline ja arusaadava dokumentatsiooniga. Lisaks kasutatakse BootstrapVue-d, mis ühendab Vue ja Bootstrap-i ning võimaldab lihtsustada kasutajaliidese arendamist valmis komponentide ja erinevate seadmete jaoks kohandumise näol [37] [38].

3.4 Koodihalduskeskkonna valik

Versioonihaldustarkvarana kasutatakse üldtuntud Git-i. Koodihalduskeskkondadest vaadeldakse kolme kõige populaarsemat Git-i toetavat tasuta keskkonda: GitLab, GitHub ja Bitbucket [39].

GitLab on avatud lähtekoodiga koodihalduskeskkond. Sellel on üle 3000 arendaja ja üle kahe miljoni kasutaja. GitLab sisaldab endas tervet DevOps elutsüklit [40]. GitHub on koodihalduskeskkond, millel on üle 56 miljoni arendaja ja üle 100 miljoni repositooriumi [41]. Bitbucket on keskkond koodi haldamiseks ja koostööks. See on kergesti integreeritav teiste Atlassian toodetega nagu Jira ja Trello, kuid mida antud töös vaja pole [42]. Antud töös kasutatakse koodi hoidmiseks Bitbucket-it, kus avalikustatakse valmis kood.

3.5 Integreeritud arenduskeskkonna valik

Java rakenduste arendamiseks kasutatakse peamiselt kolme erinevat integreeritud arenduskeskkonda: Eclipse, NetBeans ja IntelliJ IDEA [43].

Eclipse avaldati aastal 2001 IBM projektina. See on tasuta saadaval nii töölaua rakendusena kui ka veebipõhise versioonina. Eclipse sisaldab kompilaatorit, mis laeb faili iga muutuse järel koodis ning seeläbi kuvab jooksvalt probleeme. Sellel on väga palju erinevaid laiendeid ning toetatakse mitmeid erinevaid programmeerimiskeeli, raamistikke ja servereid. Eclipse oskab genereerida tavapäraseid koodi mustreid ja organiseerida sõltuvuste lisamist. Eclipse-i probleemideks võib pidada aeglust ja vaadete keerulisust, mis on tingitud sellest, et asju saab teha mitut moodi. Lisaks võib juhtuda, et laiendid ei sobi omavahel [43] [44].

NetBeans avaldati aastal 1997 ning algas tudengite projektist Prahas. See on tasuta saadaval ning ametlik integreeritud arenduskeskkond Java 8 jaoks. See tuvastab probleeme trükkimise ajal, täidab koodi automaatselt ja pakub vihjeid parandusteks. NetBeans toetab Mavenit ja Gradle jaoks on olemas laiend. Sellel on ka väga mugav analüüsi tööriist, mille abil näeb protsessori ja mälu kasutust ning aitab leida mälulekkeid [43] [44].

IntelliJ IDEA avaldati aastal 2001. See on saadaval kahes erinevas versioonis – lisavõimalustega tasuline versioon ja tasuta versioon. Tudengitele on tasulise versiooni kasutamine samuti tasuta. Tasuta versioon sobib eelkõige Java virtuaalmasinate ja Androidi arenduseks. See toetab näiteks Androidi, Javat, Kotlinit, Gradlet, Mavenit ja Git-i. Tasuline versioon sobib hästi veebiarenduseks. Tasuline versioon toetab lisaks tasuta versioonile veel näiteks JavaScripti ja TypeScripti, Java EE-d, Spring-i, andmebaasi tööriistu ja SQL-i. Lisaks tavapärastele Java redaktorile sisaldab IDEA tarka automaatset täitmist, staatiliste liikmete loomist ja sõltuvuste lisamist. See suudab ka sõne seest leida fragmente SQL-ist, CSS-ist, HTML-ist ja JavaScriptist. Lisaks leiab IDEA korduvaid koodijuppe ning võimaldab nende korraga muutmist. IDEA sisaldab ka ehituse tööriistu ja terminali akent [43] [44].

Töö autoril on kogemust IntelliJ IDEA kasutamisega ja selle võimalused sobivad väga hästi ka plaanitava rakendusega. Tudengitele on selle tasulise versiooni kasutamine tasuta, seega antud töös kasutatakse serveripoolse rakenduse arenduseks IntelliJ IDEA-t.

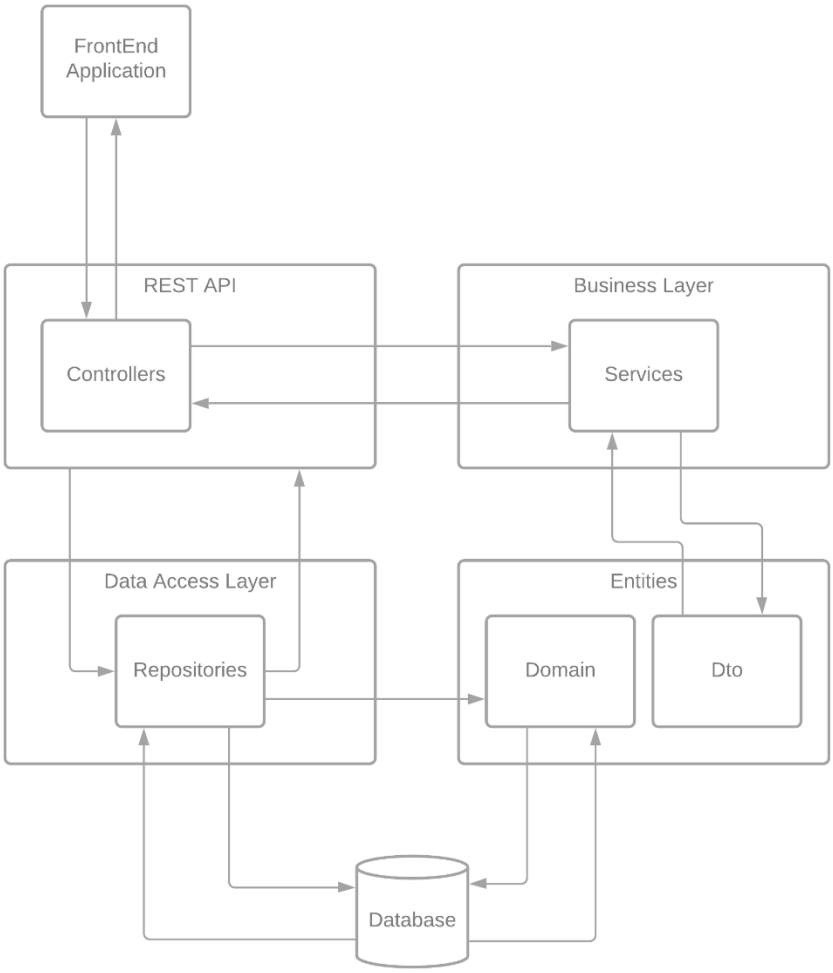
Kliendipoolse rakenduse arendamiseks kasutatakse kõige populaarsemat tasuta integreeritud arenduskeskkonda kliendipoolsete rakenduste arendamiseks - Visual Studio Code [45]. See toetab TypeScripti, võimaldab süntaksi järgi värvimist ja automaatset täitmist. See on integreeritud ka Git-i ja laiendi abil raamistik Vue-ga, mida plaanitakse antud töös kasutada [46].

3.6 Veebirakenduse arhitektuur

Plaanitav veebirakendus jaotub kahte osasse: serveripoolne rakendus ja kliendipoolne rakendus. Serveripoolne rakendus vastutab andmebaasiga suhtluse eest ja kliendipoolne rakendus teeb andmete saamiseks päringuid serveripoolsele rakendusele.

Serveripoolse rakenduse arhitektuur on välja toodud joonisel 1 ning koosneb järgnevatest kihtidest:

- Andmemudeli kiht (*Entities*) – olemid (*Domain*) ja andmesaateobjektid (*DTO – Data Transfer Object*) suhtluseks ÕIS-iga.
- Andmetele juurdepääsu kiht (*Data Access Layer*) – andmete haldus repositooriumide liideste kaudu (*Repositories*).
- Äriloogika kiht (*Business Layer*) – äriloogika kiht, kus tehakse päringuid ÕIS-i ruumide vabade aegade saamiseks ja broneerimiseks.
- REST API kiht – sisaldab MVC (*Model-View-Controller*) kontrollereid, kuhu tehakse kasutajaliidese poolses rakenduses päringuid, mis teeb vastavad päringud andmetele juurdepääsu kihti ja äriloogika kihti ja tagastab saadud andmed kasutajaliidese poolsele rakendusele.

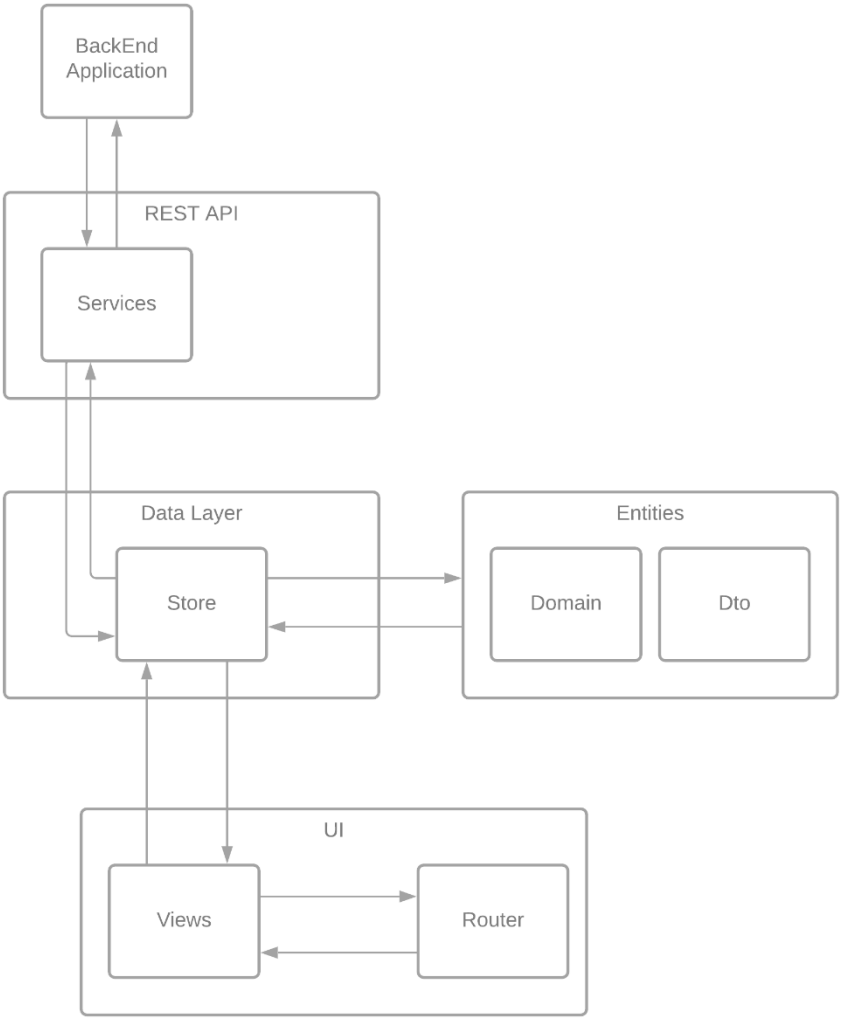


Joonis 1. Serveripoolse rakenduse arhitektuur.

Kliendipoolse rakenduse arhitektuur on välja toodud joonisel 2 ning koosneb järgnevatest kihtidest:

- Andmemudeli kiht (*Entities*) – olemid (*Domain*) ja andmesaateobjektid (*DTO – Data Transfer Object*).
- Andmete kiht (*Data Layer*) – andmete ajutine hoiustamine andmekogusse (*Store*) erinevate vaadete vahel liikudes ning suhtlus teenustega andmete uuendamiseks.
- REST API kiht – sisaldab teenuseid (*Services*), mis võtavad vastu andmete kihist tulnud kutseid ja teevad serveripoolsesse rakendusse päringuid andmete saamiseks ja muutmiseks.

- Kasutajaliidese kiht (*UI – User Interface*) – vaated (views), mida kasutaja näeb ja ruuter (*Router*) nende vaadete vahelise liikumise korraldamiseks. Teeb päringuid andmete kihti andmete saamiseks ja muutmiseks.

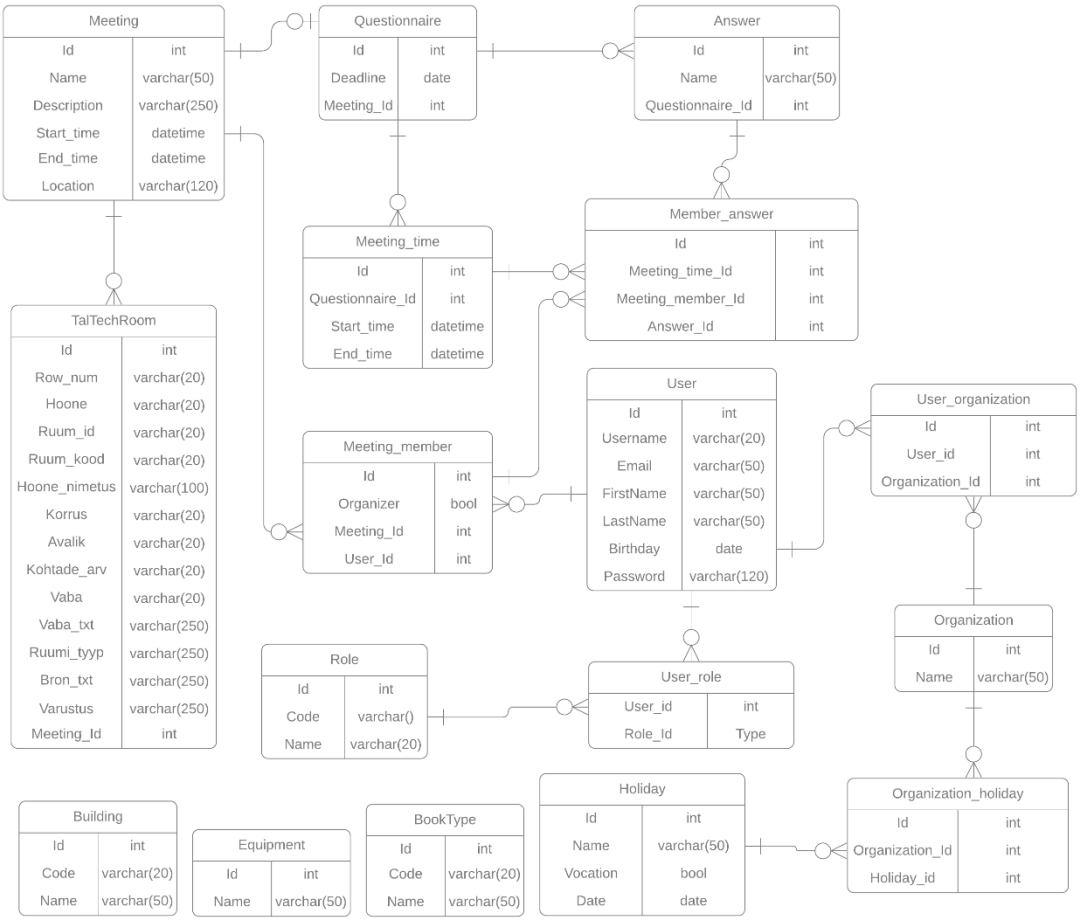


Joonis 2. Kliendipoolse rakenduse arhitektuur.

3.7 Andmebaasi mudel

Kasutajalugudest ja kasutajakogemuse disainist lähtuvalt loodi olemisuhete diagramm, mis on nähtav joonisel 3. Diagramm loodi kasutades tööriista Lucidchart. Diagrammil on näha ka kolme teistega mitte seotud tabelit: Building, Equipment ja BookType. Nende tabelite sisu saadi ÕIS-ist ning neid kasutatakse kasutajale valikute pakkumiseks ÕIS-i ruumide otsingul. Lisaks sisaldab diagramm tabelit TalTechRoom, mis sisaldab ÕIS-ist

saadud ruumide otsingust saadud ühe ruumi parameetreid. Neid parameetreid on vaja ruumide meelde jätmiseks ja hilisemaks broneerimiseks.



Joonis 3. Olemi-suhete diagramm.

3.8 Veebirakenduse disain

Antud veebirakenduse üks põhieesmärke on tudengiorganisatsioonidele kohtumiste korraldamise lihtsustamine. Kasutajakogemuse ja kasutajaliidese disainimine on suureks osaks sellest. Kasutajakogemuse ja kasutajadisaini kirjeldamine aitavad paremini läbi mõelda rakenduse funktsionaalsuse ja rõhuasetuse.

3.8.1 Kasutajakogemuse disain

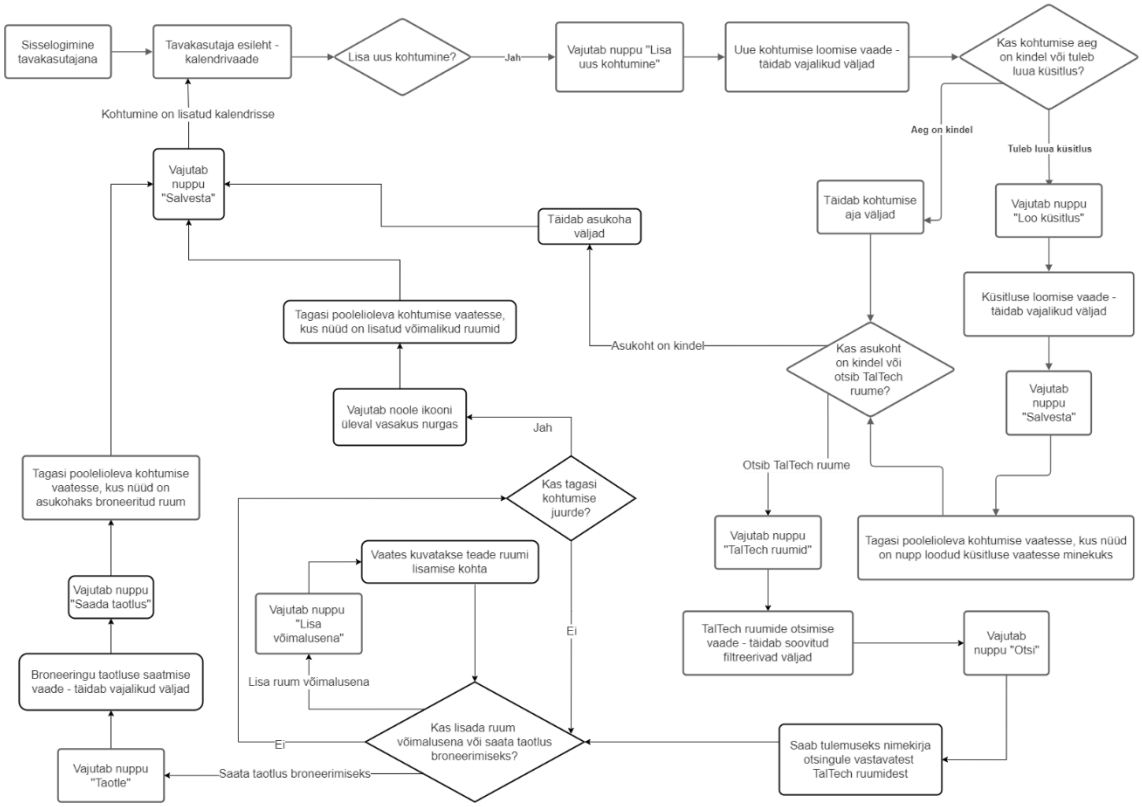
Kasutajalugudest tulenevalt võib lehe jaotada rollide järgi kaheks – kasutaja ja administraatori vaated. Administraatori vaated võib omakorda jaotada kasutajaid, organisatsioone ja tähtpäevi haldavateks lehtedeks ning need omakorda nimekirja, objekti

lisamise ja muutmise vaadeteks. Kasutaja vaated hõlmavad kalendrit, küsitluste nimekirja, küsitluste ja kohtumiste vaatamist, lisamist, muutmist ja TalTechi ruumide otsingu ning broneerimise vaateid.

Tavakasutaja esilehel on näha kohtumiste kalender, kus on võimalik filtreerida kõigi organisatsioonide, enda organisatsioonide, TalTechi ja privaatseid kohtumisi ning tähtpäevi. Tähtpäevi ise tavakasutaja lisada ei saa. Esilehel on ka nupud uue kohtumise lisamiseks ja aktiivsete küsitluste nimekirja vaatamiseks.

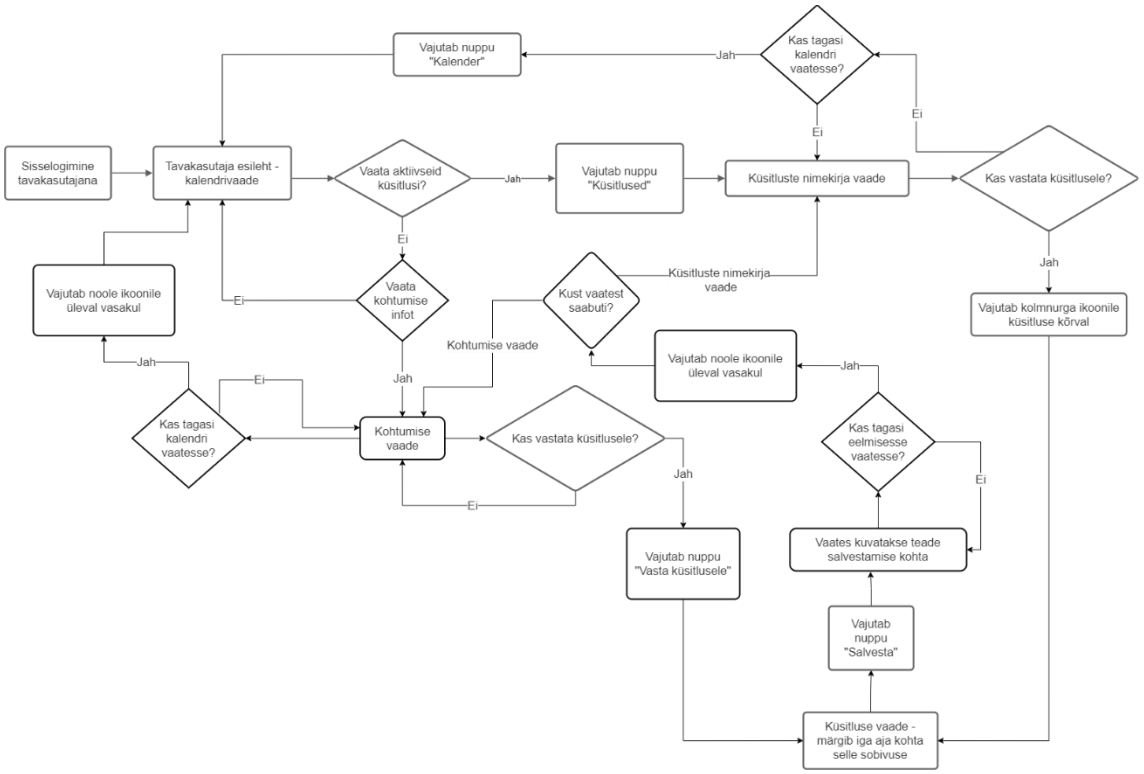
Kasutajakogemuse paremaks kujutamiseks loodi kasutajakogemuse diagrammid. Diagrammid loodi kasutades tööriista Draw.io, mis on mugav tasuta veebirakendus kasutajakogemuse diagrammide loomiseks [47].

Joonisel 4 on näha kohtumise loomise kasutajakogemus. Uue kohtumise loomisel on võimalik lisada ka küsitlus ühise aja leidmiseks ja otsida sobivaid TalTechi ruume. TalTechi ruume on võimalik nii meelde jätta hilisemaks broneerimiseks kui ka broneerimise taotluseid saata ÕIS-i.



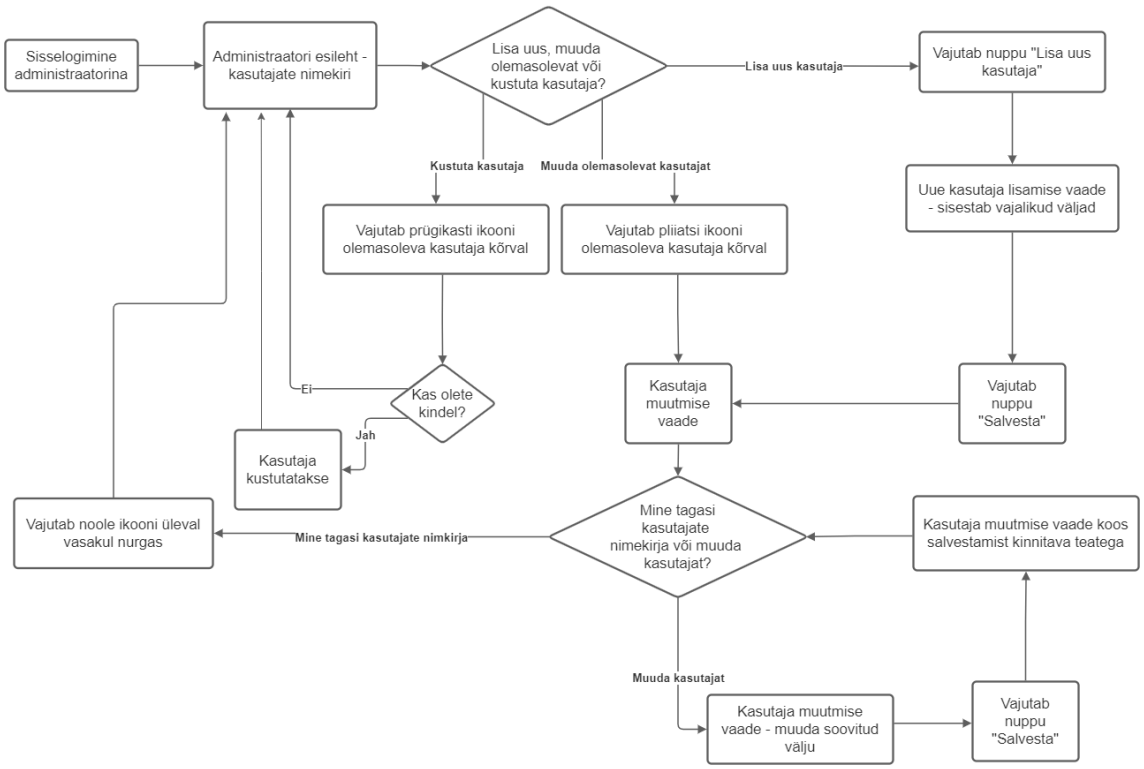
Joonis 4. Kohtumise looja kasutajakogemuse diagramm.

Joonisel 5 on näha kohtumisel osaleja kasutajakogemus. Kohtumiste infot on võimalik näha kalendris kohtumisele vajutades. Kohtumisel osalemise ja ühiste aegade leidmise küsitlustele on võimalik vastata kas kohtumise info alt või kõigi aktiivsete küsitluste nimekirjast.



Joonis 5. Kohtumisel osaleja kasutajakogemuse diagramm.

Joonisel 6 on näha administraatori vaates kasutajate lisamine, muutmine ja kustutamine. Uue tähtpäeva ja uue organisatsiooni samu protsesse eraldi välja ei tooda, sest need on oma olemuselt samad.



Joonis 6. Administraatori vaadete kasutajakogemuse diagramm.

3.8.2 Kasutajaliidese disain

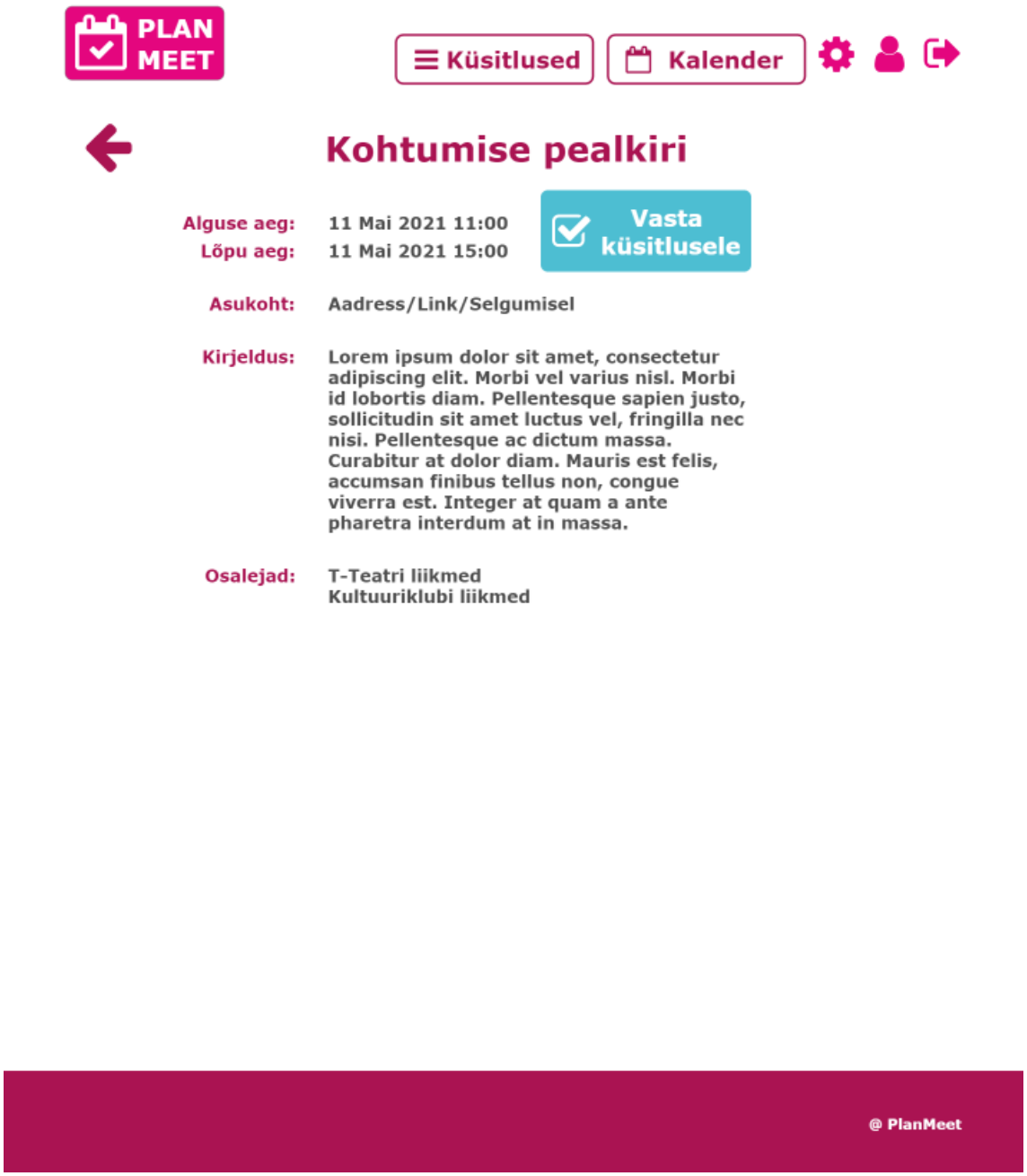
Kasutajaliidese disaini paremaks kirjeldamiseks loodi rakenduse vaadete prototüüp. Prototüübi loomiseks kasutati veebirakendust UXPin, millega autor on varem korduvalt kokku puutunud ning mis võimaldab luua interaktiivseid prototüüpe [48]. UXPin tasuta versioonist piisab antud töö jaoks.

Kasutajakogemuse põhjal sai selgeks, mis vaateid on vaja rakenduse funktsionaalsuse tagamiseks. Mugava kasutajakogemuse üheks aluseks on visuaalselt esteetiline disain ja lihtne kasutatavus. Sellele protüübi loomisel rõhutigi. Disainimisel püüti kasutada TalTechi stiiliraamatus olevaid värve ja kirjatüüpe, et sobituda olemasolevate TalTechi veebilahendustega.

Joonisel 7 on näha tavakasutaja jaoks kõige olulisem esileht – kalendrivaade. Seal on kuuvaates võimalik näha aktiivseid kohtumisi ja tähtpäevi. Kohtumisi ja tähtpäevi on võimalik rippmenüüst ka filtreerida.

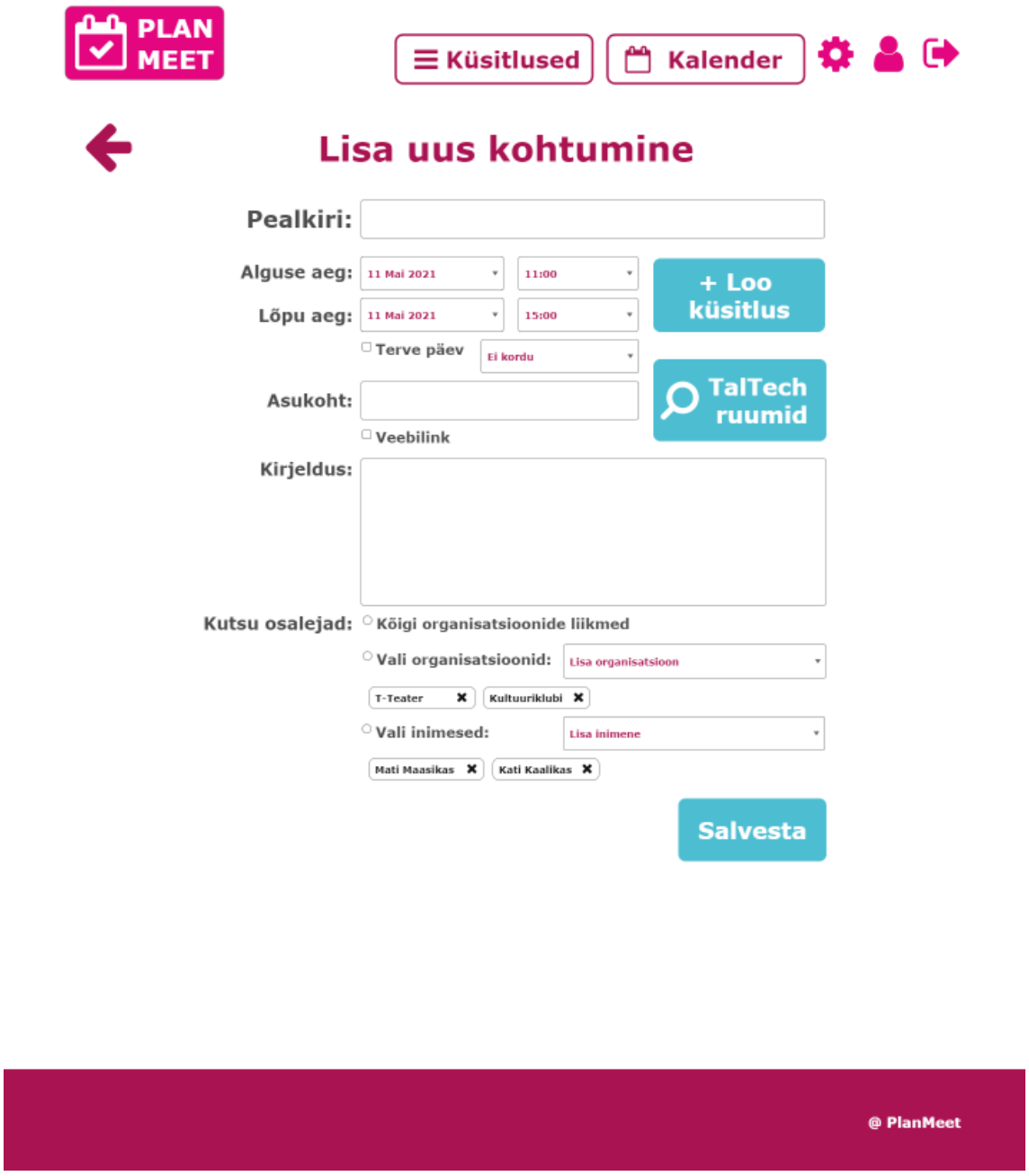
Joonis 7. Kalendrivaate disain.

Joonisel 8 on näha kohtumise info vaade, mis kuvatakse kalendris kohtumisele vajutades. Seal on näha kohtumise pealkiri, aeg, nupp osalemise küsitluse juurde minekuks, asukoht, kirjeldus ja osalejad. Kindlat aega ei pruugi kirjas olla, kui ühist aega alles otsitakse küsitluse abil.



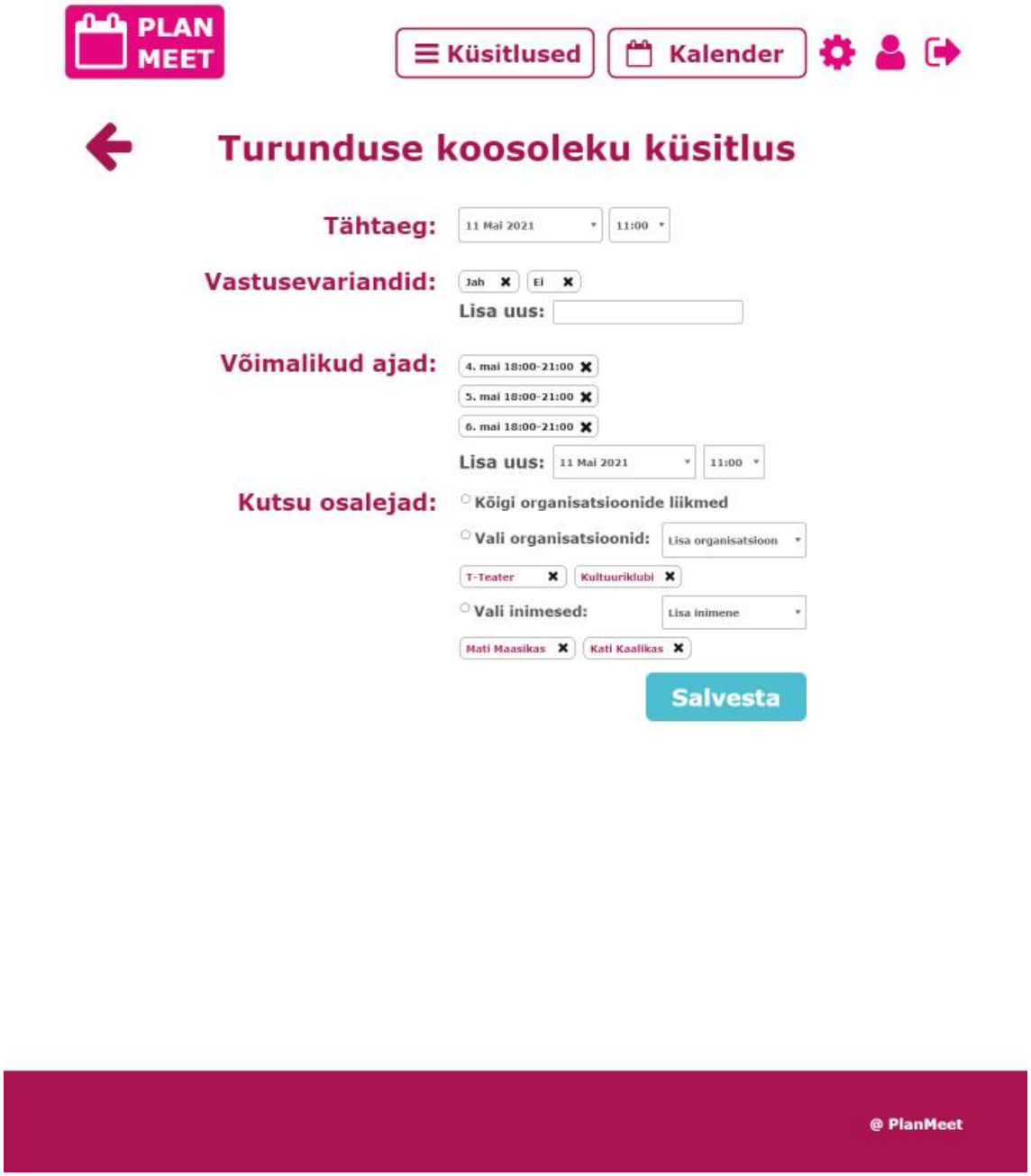
Joonis 8. Kohtumise info vaate disain.

Joonisel 9 on näha uue kohtumise loomise vaate disain. Lisada saab pealkirja, aja, asukoha, kirjelduse ja kutsuda osalejad. Aja leidmise jaoks saab ka küsitluse luua mitme võimaliku ajaga. Ürituse saab märkida ka terve päeva kestvaks või luua korduva ürituse näiteks iga päev või nädal. Asukoha määramisel saab otsida TalTechi ruume, neid meelde jätta hilisemaks broneerimiseks või taotleda. Asukohana võib lisada ka veebilingi, kui üritus toimub näiteks Zoom-is. Ürituse osalejatena saab kutsuda kas kõikide organisatsioonide liikmed, valitud organisatsioonide liikmed või valitud inimesed.



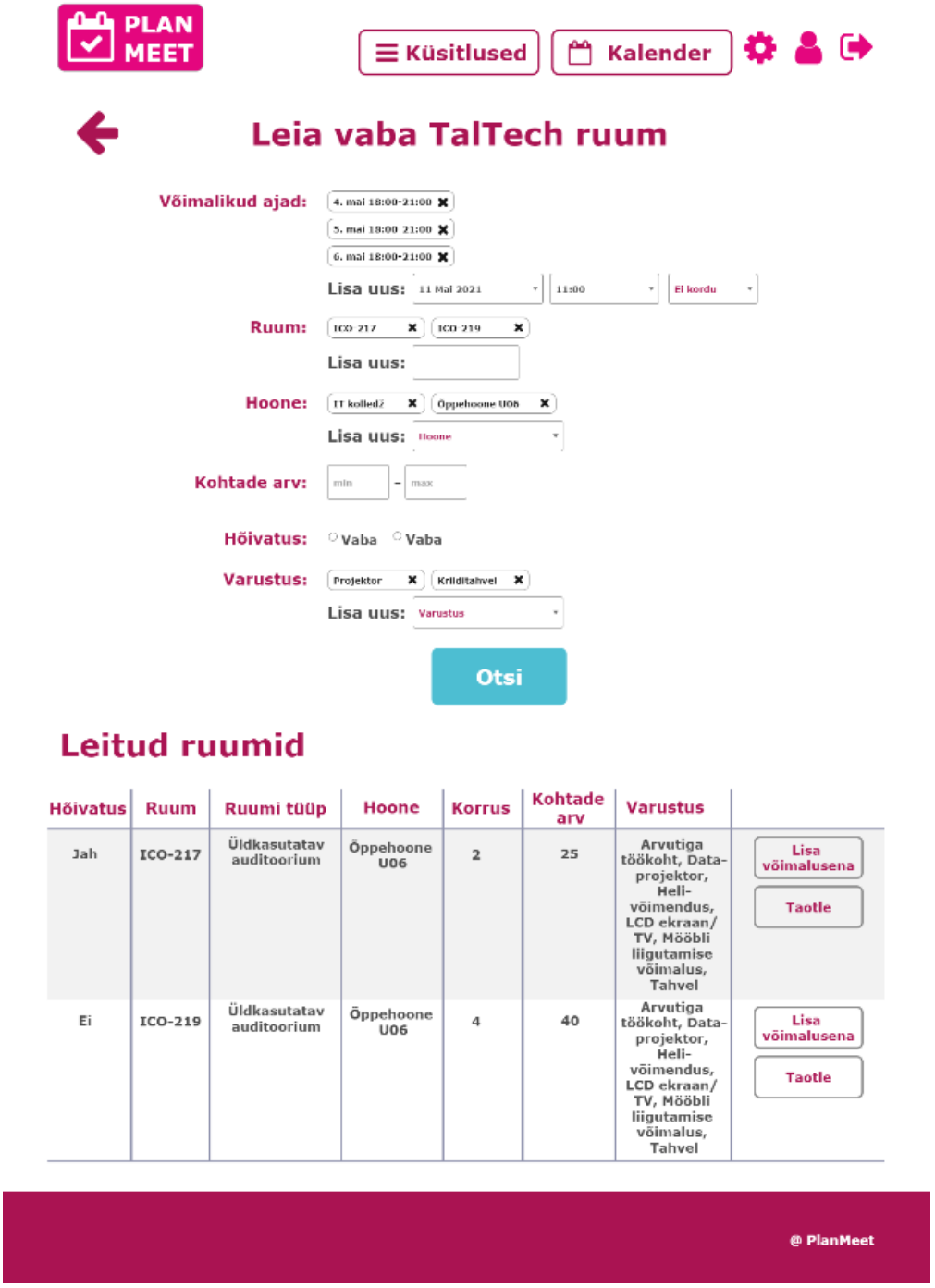
Joonis 9. Uue kohtumise loomise vaate disain.

Joonisel 10 on näha küsitluse loomise vaate disain. Seal on võimalik lisada tähtaeg küsitlusele vastamiseks, seadistada vastusevariante, märkida võimalikud kohtumise toimumise ajad ja kutsuda osalejad. Kui kohtumise juurde olid osalejad juba märgitud, siis jõuavad need ka automaatselt küsitluse vaatesse.



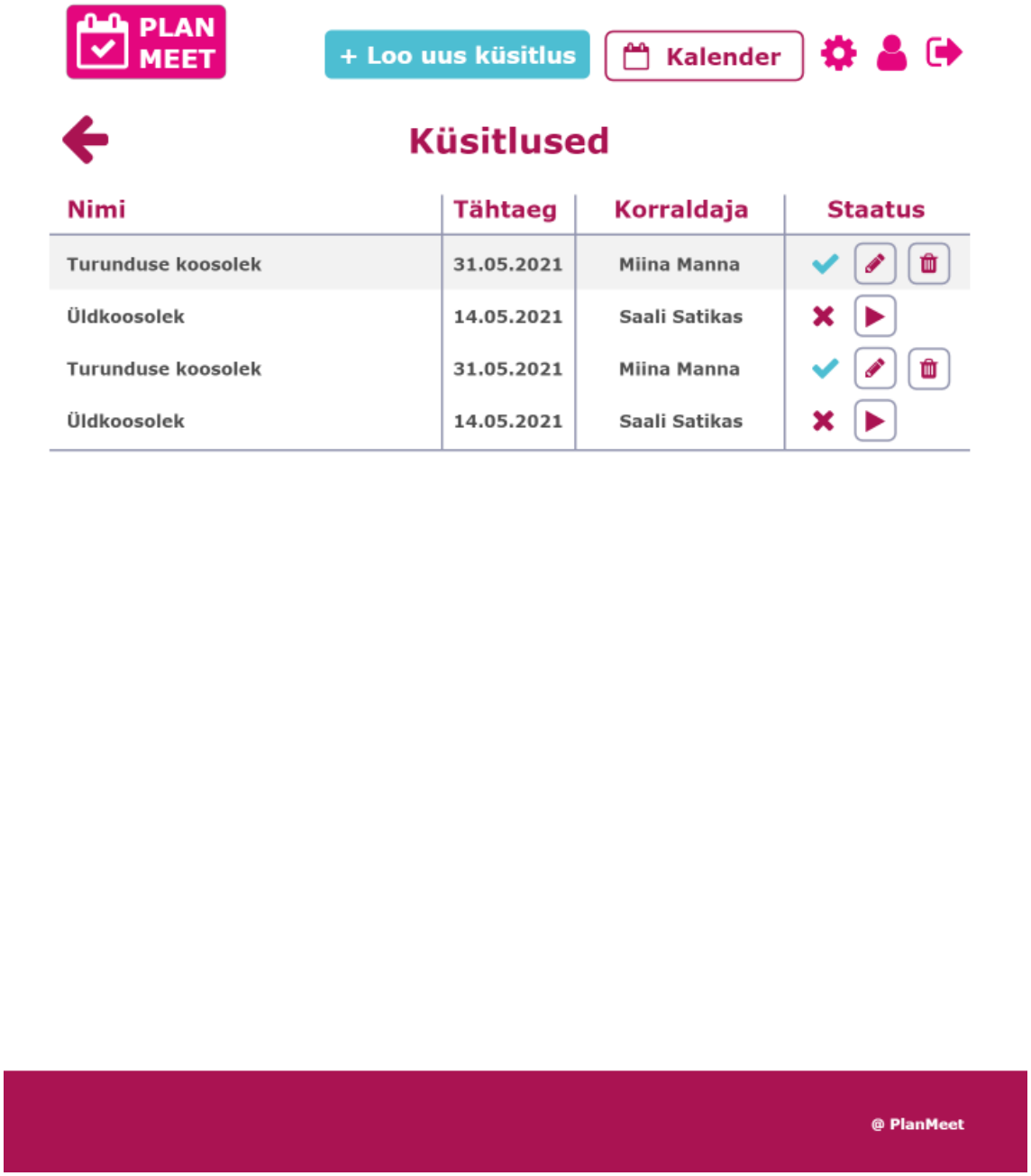
Joonis 10. Küsitluse loomise vaate disain.

Joonisel 11 on näha TalTechi ruumide otsimise vaade. Seal on võimalik määrata erinevaid filtreerivaid parameetreid nagu praegugi ÕIS-is. Erinevusena võib tuua lihtsama aegade lisamise ja mitme ruumi ja hoone järgi korraga otsimise. Päringu tulemusena tagastatakse nimekiri võimalikest ruumidest, mida on võimalik kas hilisemaks broneerimiseks meelde jätta kohtumise juurde või saata taotlus ÕIS-i ruumi broneerimiseks.



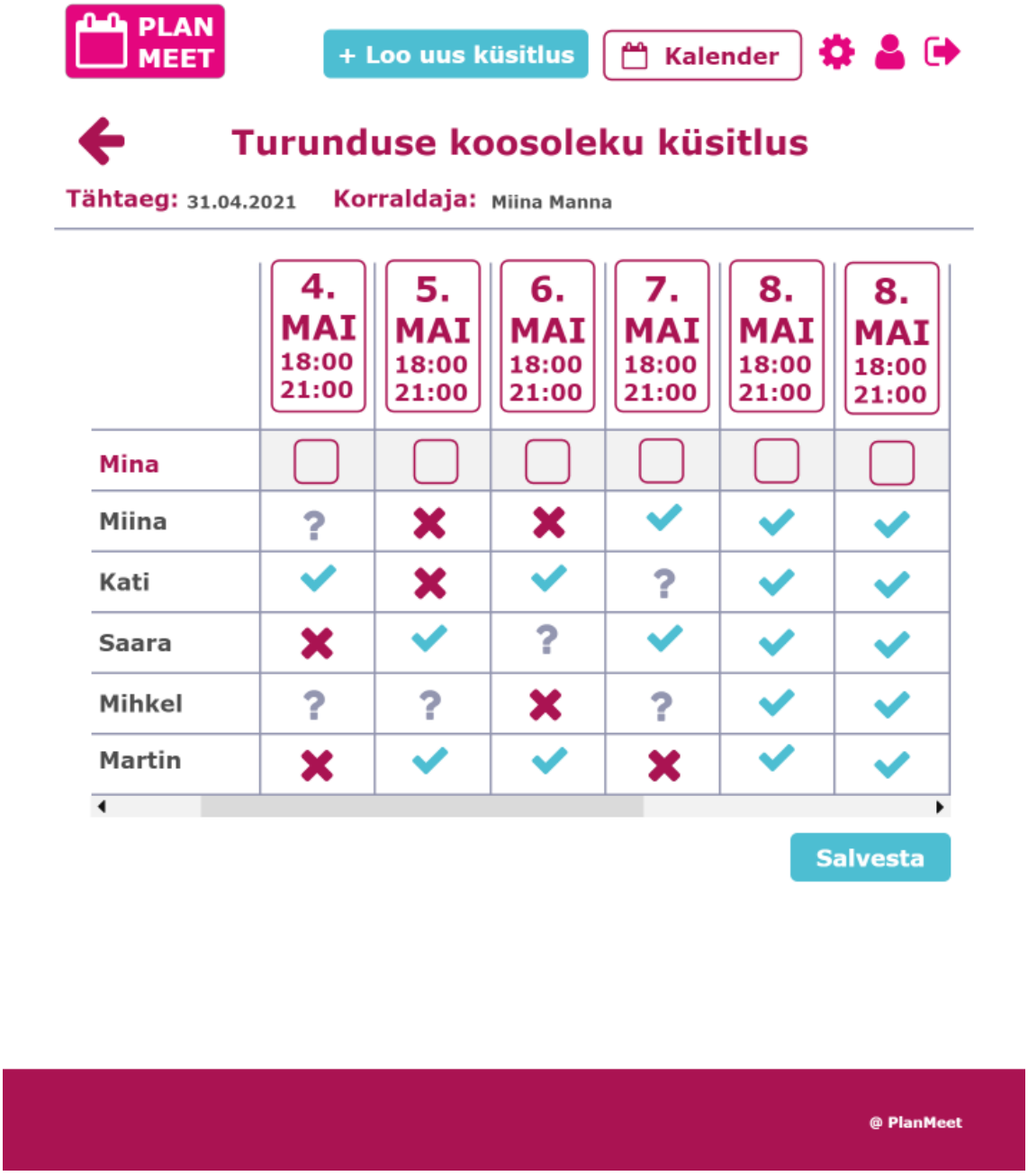
Joonis 11. TalTechi ruumide otsimise vaate disain.

Joonisel 12 on näha kõigi aktiivsete küsitluste vaade. Seal on näha kõik küsitlused, kuhu kasutajat on kutsutud või mille ta on loonud. Staatusena on märgitud, kas kasutaja on küsitlusele vastanud või mitte ning nupp vastamiseks või vastuse muutmiseks. Küsitlusi, mille omanik kasutaja on, saab ta ka muuta ja kustutada.



Joonis 12. Küsitluste vaate disain.

Joonisel 13 on näha küsitlusele vastamise vaate disain. Kasutaja saab märkida, mis ajad talle sobivad kohtumiseks, mis võib-olla sobivad ja mis ei sobi. Kasutaja näeb ka teiste osalejate vastuseid, tähtaega ja korraldajat.

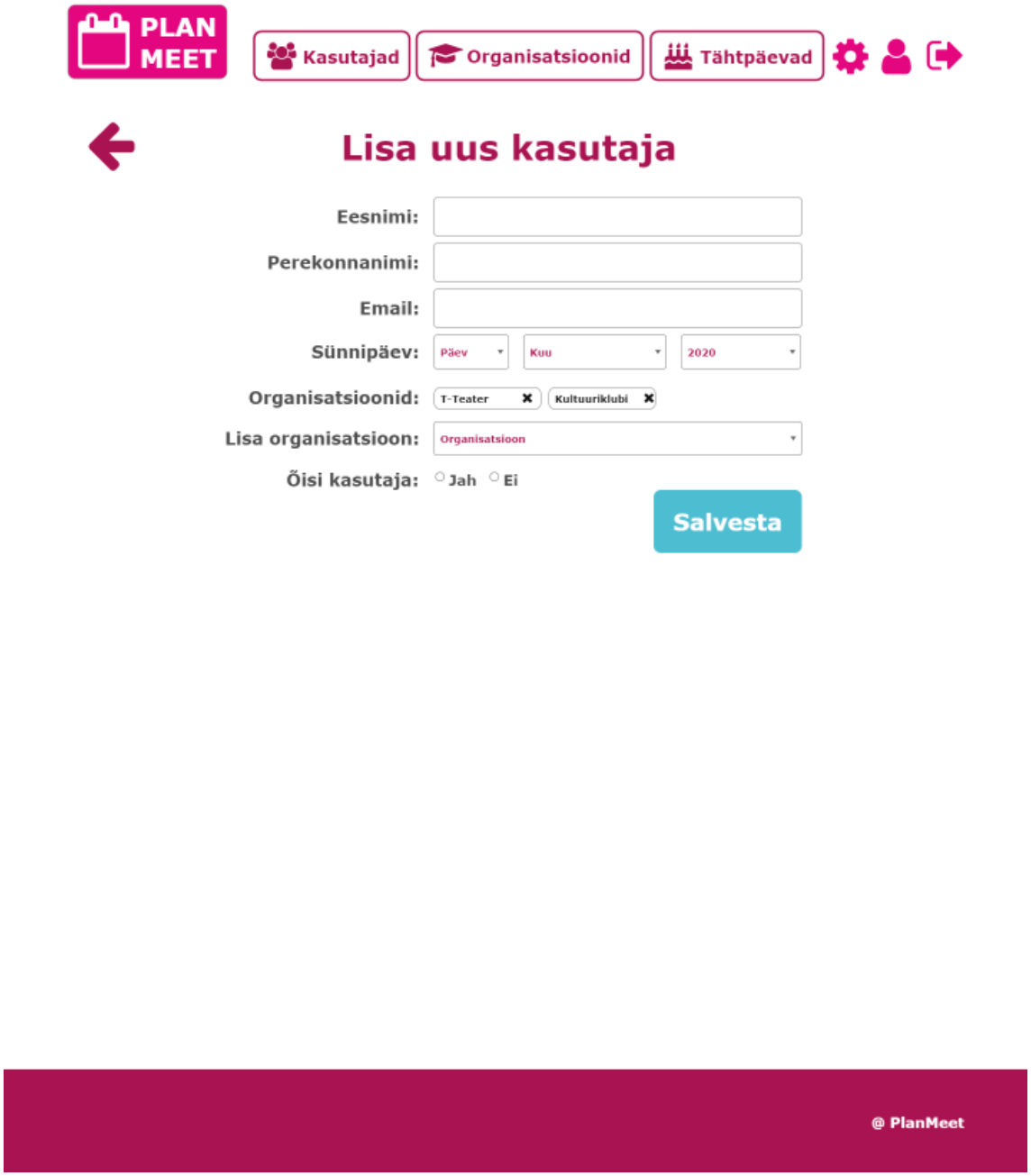


Joonis 13. Küsitlusele vastamise vaate disain.

Joonisel 14 on näha administraatori kasutajate nimekirja vaate disain. Administraator saab lisada, muuta ja kustutada kasutajaid, organisatsioone ja tähtpäevi. Prototüübis loodi vaated vaid kasutajate kohta, sest vaated organisatsioonide ja tähtpäevade jaoks on disainilt samad. Nimekirja vaates näeb esmast infot kasutaja kohta, saab kasutaja kustutada ja minna muutmise vaatesse.

Joonis 14. Administraatori kasutajate nimekirja vaate disain.

Joonisel 15 on näha administraatori kasutaja muutmise ja lisamise vaate disain. Seal on võimalik määrata kogu info kasutaja kohta nagu nimi, sünnipäev (kalendrisse tähtpäevana lisamiseks), milliste organisatsioonide liige kasutaja on ja kas tal on ÕIS-i kasutaja ehk kas ta saab ÕIS-is vabu ruume vaadata ja taotleda. Mitte ÕIS-i kasutajad saavad endiselt PlanMeet rakendust kasutada, kuid mitte TalTechi ruumide päringuid teha, sest selleks tuleb ÕIS-i sisse logida.



Joonis 15. Administraatori kasutaja muutmise ja lisamise vaate disain.

3.9 Analüüsi kokkuvõte

Analüüsi käigus sai selgeks, milline saab olema loodav rakendus – nõuded, funktsionaalsus, tehnoloogiad, arhitektuur, andmebaas ja disain. Kasutajalugudest tulenevalt loodi rakenduse nõuded, millel kõik järgnevad otsused põhinevad.

Serveripoolse rakenduse tehnoloogiateks valiti Java programmeerimiskeel, Spring Boot raamistik, Hibernate raamistik andmebaasiga suhtluse toetamiseks ja Gradle projekti

ehitamise tööriistana, mille kõigiga on autoril kogemust ning mis on populaarsed ja veebirakenduse arendust lihtsustavad võimalused. Andmebaasiks valiti MariaDb, mis on tasuta ja kergesti hallatav ning kasutatav. Kliendipoolse rakenduse jaoks kasutatakse HTML-i, CSS-i ja TypeScripti ning raamistikku Vue, mis kõik on populaarsed kaasaegsed tehnoloogiad veebirakenduste arendamiseks.

Koodihalduskeskkonnaks valiti Bitbucket integreeritud arenduskeskkondadest kasutatakse serveripoolse rakenduse arendamiseks IntelliJ IDEA-t ja kliendipoolse rakenduse arendamiseks Visual Studio Code-i.

4 Veebirakenduse arendus

Käesoleva veebirakenduse arendus on jaotatud kahte alampeatükki: serveripoolse rakenduse arendus ja kliendipoolse rakenduse arendus.

4.1 Serveripoolse rakenduse arendus

Serveripoolse rakenduse arendamisel kasutati Java programmeerimiskeelt ja raamistikku Spring Boot. Serveripoolne rakendus sisaldab andmebaasiga suhtlust, äriloogikat ja integratsiooni ÕIS-iga.

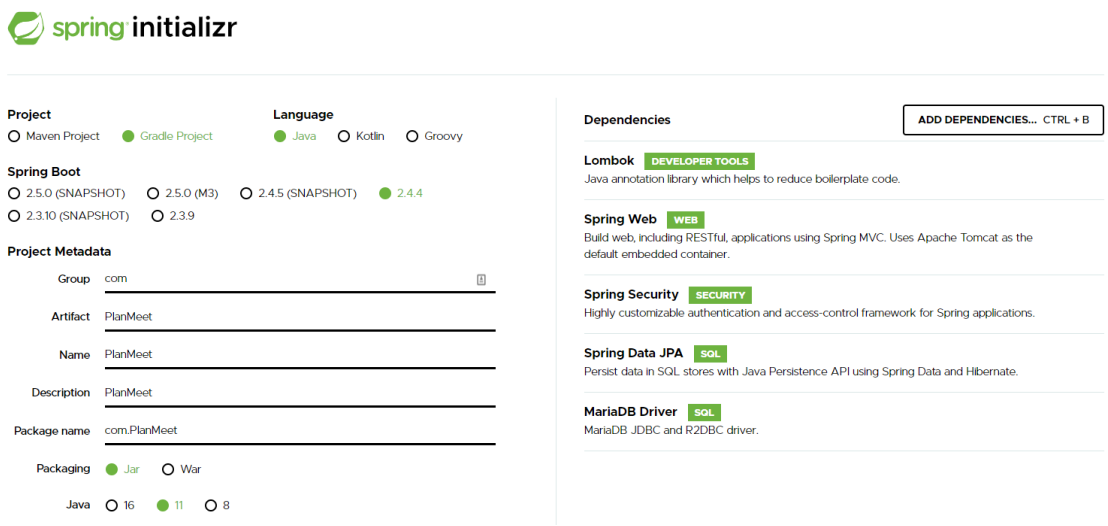
4.1.1 Rakenduse loomine

Esmane rakendus loodi kasutades Spring Initializr-it, mis võimaldab kiiresti ja lihtsalt luua Java Spring Boot rakenduse vastavalt soovitud parameetritele. Selle abil on võimalik lisada ka vajalikke sõltuvusi [49]. Joonisel 16 on näha parameetrid, mida kasutati. Projekti ehitamise tööriistana kasutatakse Gradle-t, mille kaudu lisatakse ka sõltuvused. Spring Boot-i puhul kasutatakse praegust ametlikku versiooni [50]. Rakendust on võimalik pakkida ka .jar või .war faili, aga kuna lõputöö skoobis ei ole rakenduse evitus pilve, siis pole see oluline. Java versiooniks valiti 11, sest see on kõige uuem pikaajalise toetusega versioon [51].

Spring Initializr kaudu lisati ka sõltuvused Lombok, Spring Web, Spring Security, Spring Data JPA ja MariaDb Driver. Lombok võimaldab vähendada koodiridu andmeobjektide loomisel [52]. Spring Web võimaldab REST rakenduste loomist MVC-arhitektuuri kasutades. Spring Security abil on võimalik lisada rakendusele turvakiht ning kasutajad. Spring Data JPA kaudu on võimalik vältida lisatööd andmebaasi päringute tegemisel ning tabeleid luua, sealt andmeid saada ning sinna andmeid lisada automaatselt väheste koodiridadega. MariaDb Driver on vajalik JPA automatiseerimise võimaldamiseks.

Spring Initializr kaudu luuakse .zip fail, mida on võimalik lahti pakkida ja soovitud integreeritud arenduskeskkonnas avada. PlanMeet serveripoolse rakenduse arendamisel kasutati integreeritud arenduskeskkonda IntelliJ IDEA.

Spring Initializr genereerib rakenduse esmase struktuuri koos vajalike sõltuvustega. See hõlmab endas näiteks *build.gradle* faili, kus lisatakse vajalikud pistikprogrammid, sõltuvused ja nende versioonid ning *.gitignore* faili, kus määratakse, mis laetakse üles koodihalduskeskkonda ja mida mitte. Luuakse ka konfiguratsiooni fail *application.properties* ja rakenduse peamine klass, kust rakendust käivitatakse. Rakenduse peamisele klassile lisatakse annotatsioon *@SpringBootApplication*, mis hõlmab endas annotatsioone *@Configuration*, *@EnableAutoConfiguration* ja *@ComponentScan*. Annotatsioon *@Configuration* määrab selle klassi aluseks, mille järgi otsida rakenduse erinevaid komponente. Annotatsioon *@EnableAutoConfiguration* annab Spring Boot-ile märku, et ta otsiks ise erinevaid komponente klassiraja (*classpath*) seadete alusel. Annotatsioon *@ComponentScan* annab Spring-ile märku, et ta otsiks erinevaid annotatsioonidega märgistatud komponente rakenduse sellest kaustast, kus peamine klass paikneb [53].



Joonis 16. Spring Initializr parameetrid.

4.1.2 Andmebaasi loomine

Andmebaasiks valiti MariaDb kõige uuema võimaliku versiooniga ning selle halduseks kasutatakse HeidiSQL-i. HeidiSQL on avatud lähtekoodiga Windowsile sobiv kasutajaliides SQL andmebaasidele [54]. Seal loodi andmebaas PlanMeet. Rakenduse andmebaasiga ühendamiseks kasutatakse MariaDb Driver-it ja Spring Data JPA-d, mis lisati Spring Initializr-is. Spring Data JPA võimaldab andmebaasi tabeleid luua otse domeeni objektidest.

Andmebaasi tabelite loomiseks JPA abil tuleb andmebaasi ühenduse parameetrid lisada konfiguratsiooni faili *application.properties* ja domeeni objektidele lisada JPA annotatsioon *@Entity*, et märgistada JPA jaoks domeeni objektid . Samuti tuleb määrata peamine unikaalne võti annotatsiooniga *@Id*. Lisaks tuleb peamisele võtmele lisada annotatsioon *@GeneratedValue(strategy = GenerationType.IDENTITY)*, mis genereerib võtmeid automaatselt. Igale domeeni objektile tuleb JPA jaoks lisada ka ilma parameetriteta konstruktor. Äriloogika jaoks lisatakse soovitud parameetritega konstruktorid, mille abil uusi objekte luuakse [53]. Domeeni objektidele lisatakse ka Lombok raamistiku annotatsioonid *@Setter*, *@Getter*, *@ToString* ja *@NoArgsConstructor*, mis hoiavad kokku koodiridu ja loovad need funktsioonid automaatselt. Tabelite ühendamiseks kasutatakse annotatsioone *@OneToMany(mappedBy = "<välja nimi>", cascade = CascadeType.ALL)* ja *@ManyToOne*. Märge *cascade = CascadeType.ALL* tähendab seda, et kui ülemobjekt kustutatakse, siis kustutatakse ka temaga seotud alamobjektid.

Andmebaasi objektide lisamiseks, muutmiseks, kustutamiseks ja pärimiseks tuleb rakendusele lisada iga domeeni objekti jaoks hoidla (*repository*). Kõik hoidlad laiendavad Spring Data liidest *CrudRepository*, millele antakse parameetrina domeeni objekt ja võtme tüüp. Sellised hoidlad võimaldavad JPA kaudu automaatselt käivitamise ajal luua hoidla funktsioonid andmete lisamiseks, muutmiseks, kustutamiseks ja pärimiseks. Ilma JPA-ta tuleks need funktsioonid käsitsi kirjutada. Võimalik on lisada ka oma funktsioone. Näiteks on võimalik objekte küsida mõne konkreetse välja järgi [53].

JPA kasutamine toimub konfiguratsiooni failis oleva info järgi automaatselt. Konfiguratsiooni failis määratakse, kas JPA kustutab käivitamise ajal kogu olemasoleva andmebaasi sisu ja loob uuesti, uuendab või lihtsalt kontrollib, et sisu vastaks nõudmistele [53]. Lisaks loodi *data.sql* fail, kuhu kirjutati SQL laused esmaste andmete andmebaasi lisamiseks.

4.1.3 REST API

Serveripoolse ja kliendipoolse rakenduse vaheliseks suhtluseks kasutatakse REST API-t. Serveripoolses rakenduses loodi selleks REST kontrollerid annotatsiooniga *@RestController*. Igale kontrollerile määratakse oma aadress annotatsiooniga *@RequestMapping(„/<aadress>“)*. Kontrollerid sisaldavad erinevaid funktsioone andmete saamiseks, lisamiseks, muutmiseks ja kustutamiseks. Igal funktsioonil on vastav

annotatsioon *@GetMapping(„/<aadress>“)*, *@PostMapping(„/<aadress>“)* või *@DeleteMapping(„/<aadress>“)*. Päringuid saab teha ka parameetritega, näiteks id järgi objekti küsimiseks saab kasutada annotatsiooni *@GetMapping("/{id}")* ja funktsiooni parameetrit *@PathVariable Integer id*. Objekti lisamisel antakse objekt kaasa päringu kehas (*Request body*). Objekti lisamisel kasutatakse annotatsiooni *@PostMapping("")* ja funktsiooni parameetrina *@RequestBody <Objekti tüüp> <objekti nimetus>*. Kontrolleri funktsioonides tehakse päringuid repositooriumidesse, mis suhtlevad andmebaasiga andmete saamiseks või muutmiseks.

4.1.4 Kasutajate ja kasutajagruppide funktsionaalsus ning turvalisus

PlanMeet kasutajate haldamiseks ja turvalisuse tagamiseks kasutatakse Spring Security-t ja JWT (*JSON Web Token*) autentimist. See tähendab, et sisse logimisel saab kasutaja JWT tokeni, mis igale päringule kaasa antakse kuni välja logimise ja koodi eemaldamiseni. Token saadetakse HTTP päringu päises. Server valideerib tokeni ja tagastab selle sobivuse korral soovitud vaste. Token salvestatakse kliendi pool veebilehitseja lokaalses mälus. [55]

Kasutajate jaoks loodi kaks API aadressi *@PostMapping("/signup")* ja *@PostMapping("/signin")* uue kasutaja lisamiseks ja kasutaja sisse logimiseks. Uue kasutaja loomisel küsitakse kasutajanime, e-maili, eesnime, perekonnanime, sünnipäeva, parooli, nimekirja rollidest ja organisatsioonidest, mille õigused kasutajale lisatakse. Tavakasutajatele antakse roll „*ROLE_USER*“ ja administraatoritele roll „*ROLE_ADMIN*“. Õiguste erinevused seisnevad API aadressides, millele roll ligi saab. Administraator saab lisaks tavakasutajale ligi kasutajate, organisatsioonide ja tähtpäevade muutmisele, kustutamisele ja lisamisele. Ilma kasutajata saab ligi vaid sisse logimise aadressile, kõige muu jaoks peab vähemalt tavakasutaja olema.

Kasutajate lisamine toimub administraatori poolt, kes peab enne kontrollima isiku kuuluvust organisatsiooni. Kasutaja lisamisel kontrollitakse, et sama kasutajanime ja e-mailiga kasutajat juba olemas poleks ning kui kõik andmed sobivad, siis lisatakse kasutaja andmebaasi ja tagastatakse teade kasutaja eduka lisamise kohta. Kui mõni parameeter ei sobi, siis tagastatakse sõnum tekkinud probleemi kohta. Kasutaja parooli ei salvestata otse, vaid kodeeritakse Spring Security liidese *PasswordEncoder* kaudu. Sisse logimisel võrreldakse sisestatud parooli kodeeritult andmebaasis oleva kodeeritud parooliga. Kasutaja sisse logimiseks küsitakse kasutajanime ja parooli. Kui vastava kasutajanime ja

parooliga kasutaja leitakse, siis genereeritakse JWT token ning tagastatakse koos kasutaja ülejäänud infoga.

4.1.5 Integratsioon ÕIS-iga

ÕIS-iga ühendumiseks serveripoolsest rakendusest tehakse päringuid Java HttpClient-i abil. Vabade ruumide leidmiseks tuleb esmalt ÕIS-i sisse logida, teha üks POST päring soovitud otsingu parameetritega ja siis GET päring ruumide nimekirja saamiseks JSON-i kujul. Peale soovitud andmete saamist logitakse ka välja.

ÕIS-ist andmete saamiseks tuleb kasutada olemasolevat ÕIS-i kasutajat. Rakenduse kasutamiseks lõputöö skoobis tuleb serveripoolse rakenduse *application.properties* faili kirjutada oma ÕIS-i kasutajanimi ja parool. Edasiarendusena tulevikus peaks ÕIS-i sisse logimise tegema osaks rakenduse töövoost. Peale sisse logimist tuleb salvestada saadud küpsis (*Cookie*), mis lubab edasisi tegevusi sooritada. Küpsiste salvestamiseks kasutatakse Java CookieManager-i, mis küpsiseid salvestab ja päringutele kaasa annab.

Ruumide otsimisel saab kaasa anda parameetrid ruum, hoone, korrus, minimaalne ja maksimaalne kohtade arv, kas ruum peab olema avalik või mitte ja kas ruum peab olema vaba või mitte. Lisaks saab anda mitu aega ja varustuse. Peale päringu postitamist saab teha päringu ruumide JSON-i saamiseks. Tagastatakse sobivate ruumide arv ja nimekiri ruumidest koos erinevate parameetritega nagu ruum, hoone, korrus avalik või mitte, vaba või mitte, ruumi tüüp ja varustus. Samuti saadakse kood, mille abil on võimalik saata taotlus ruumi broneerimiseks.

PlanMeet rakenduses ruumide broneerimise üheks eeliseks on mugavam kasutus. Selle üheks väljundiks on see, et ruumide otsingut on võimalik ka mitme ruumi ja hoone järgi korraga teha. Selleks tehakse serveripoolses rakenduses mitu päringut ja pannakse tulemused kasutajale vaatamiseks kokku. Nii tehakse korduvad otsingud, mida muidu pidi käsitsi tegema, kasutaja eest ära.

Ruumi broneerimiseks tehakse kõigepealt päring vastava koodiga, mis ruumide nimekirjast saadi ja siis broneerimise päring koos broneerimise tüübi koodi, kommentaari ja isiku koodiga, kelle nimele ruum broneeritakse. Selle tulemusena saadetakse ruumi broneerimise taotlus.

4.2 Kliendipoolse rakenduse arendus

Kliendipoolse rakenduse arendamisel kasutati TypeScript, HTML ja CSS programmeerimiskeeli ning raamistikku Vue. Kliendipoolne rakendus sisaldab suhtlust serveripoolse rakendusega ja kasutajaliidese vaateid.

4.2.1 Rakenduse loomine

Kliendipoolne rakendus loodi kasutades Vue CLI-d. See on käsurea põhine lahendus, mis võimaldab vastavalt soovitud parameetritele kiirelt ja lihtsalt Vue esmase rakenduse luua. See laeb alla ja seadistab vajalikud tööriistad ning loob vajalikud komponendid vältides kõige nullist alustamist. [56] Joonisel 17 on näha loodud rakenduse seadistus. Rakenduse loomisel lisati paketid Babel, TypeScript, Router, Vuex ja Linter. Babel on JavaScripti kompilaator [57]. TypeScript pakett võimaldab projekti TypeScript-is kirjutada. Routeri all mõeldakse Vue Router-it, mis võimaldab lehtede vahel liikumist [58]. Vuex on oleku halduse muster/raamistik, mis võimaldab rakenduse olekut hallata [59]. Linter võimaldab vaadete mallid ja skripti ühte komponenti lisada [60]. Vue versiooniks valiti 2, mis on uusim täielikult stabiilne versioon [61]. Järgnevate valikute puhul valiti soovituslikud vaikimisi võimalused. Vue CLI kaudu loodud projekt sisaldab valitud raamistike sõltuvusi ja seadistusi. Samuti loodi esmane rakenduse ülesehitus

```
Vue CLI v4.5.12
? Please pick a preset: Manually select features
? Check the features needed for your project: Choose Vue version, Babel, TS, Router, Vuex, Linter
? Choose a version of Vue.js that you want to start the project with 2.x
? Use class-style component syntax? Yes
? Use Babel alongside TypeScript (required for modern mode, auto-detected polyfills, transpiling JSX)? Yes
? Use history mode for router? (Requires proper server setup for index fallback in production) No
? Pick a linter / formatter config: Standard
? Pick additional lint features: Lint on save
? Where do you prefer placing config for Babel, ESLint, etc.? In dedicated config files
? Save this as a preset for future projects? No
```

Joonis 17. Vue rakenduse seadistus.

4.2.2 Rakenduse ülesehitus

Vue CLI abil loodud projekti ülesehitus koos vajalike täiendustega:

- Node_modules – kaust kõigi alla laetud pakettide jaoks.
- Public – sisaldab rakenduse tunnusikooni (*favicon*).
- Src – projekti põhikaust
    - Assets – sisaldab rakenduse logo.

- Domain – andmesaate objektid kliendipoolses rakenduses.
- Fonts – rakenduses kasutatavad kirjatüübid.
- Router – rakenduse vaadete vahel liikumiseks vajalik ruuter, kus registreeritakse kõik vaadete aadressid.
- Services – teenused serveripoolse rakendusega suhtlemiseks.
- Store – rakenduse oleku haldamiseks mõeldud hoidla.
- Views – rakenduse vaated.
- App.vue – rakenduse juur, mis määrab rakenduse peamise malli.
- Main.ts – TypeScript fail, mis initsialiseerib App.vue ja seab üles lisamoodulid
- Shims-tsx.d.ts – aitab intergreeritud arenduskeskkonnal tuvastada .tsx faile ja lubab jsx süntaksi.
- Shims-vue.d.ts – aitab integreeritud arenduskeskkonnal tuvastada .vue faile.

- Rakenduse juurkaustas on ka konfiguratsioonifailid:
  - Tsconfig.json – märgib ära TypeScript projekti juurkausta ja kompilaatori seadistuse projekti kompileerimiseks [62].
  - README.md – tekstifail, mis tutvustab ja seletab projekti [63].
  - Package.json – projekti metaandmed ja sõltuvused npm-i jaoks [64].
  - Package-lock.json – sisaldab endas pakettide täpseid versioone, mis võimaldavad identse projekti luua [65].
  - Babel.config.js – Babel-i seadistus.
  - .gitignore – sisaldab infot failide kohta, mida Git ignoreerib ning ei lae üles versioonihalduskeskkonda [66].

- .eslintrc.js – Linter-i seadistus.
- .editorconfig – arenduskeskkonna seadistus, mis võimaldab hoida ühtset stiili [67].
- .browserslistrc – seadistus veebilehitsejate ja Node.js versioonide jaoks [68].

4.2.3 Suhtlus serveripoolse rakendusega

Serveripoolse rakendusega suhtluseks tehakse päringuid serveripoolse rakenduse REST API kontrollerites olevatele aadressidele. Päringute tegemiseks kasutatakse Axios raamistikku. Axiost on üks populaarseim HTTP klient. Teine võimalus selle kõrval on Fetch, aga Fetch-il pole automaatset JSON-i parsimist ja Axios-e süntaks päringute tegemiseks on lihtsam. Axiose üks põhimõte on vastuse ootamine. Peale päringu tegemist oodatakse kindlas vormis vastust [39].

Kliendipoolses rakenduses toimub päringute saatmine kaustas *services*, kus on abstraktsed klassid, mis sisaldavad funktsioone andmete saamiseks, lisamiseks, muutmiseks ja kustutamiseks. Iga päringu puhul kontrollitakse saadud vastust ning vastuse staatust.

4.2.4 Rakenduse oleku haldamine

Rakenduse oleku haldamiseks rakenduse siseselt kasutatakse Vuex raamistikku. Vuex on raamistik/muster, mida kasutatakse Vue rakenduste staatuse haldamiseks. Vuex võimaldab keskse hoidla (*store*) kõigile rakenduse komponentidele. Ühine hoidla tagab oleku muutumise vaid etteaimataval viisil ning väldib probleeme, mis võivad tekkida, kui mitu vaadet muudavad korraga sama objekti [59].

Vuex hoidla sisaldab endas olekut (*state*), saajaid (getters), mutatsioone (*mutations*), tegevusi (*actions*) ja mooduleid (*modules*). Olekuks on terve rakenduse peale üks oleku puu, mis koosneb kõigist komponentide poolt kasutatavatest objektidest. Saajad on juhtudeks, kui on vaja olekust arvutatud parameetreid. Mutatsioonide kaudu toimub oleku muutmine. Selleks tuleb mutatsioon hoidlasse saata (*commit*-ida) kujul *store.commit('<mutatsiooni nimetus>', <mutatsiooniga kaasa minev parameeter>)*. Tegevused on mutatsioonidega sarnased, kuid oleku muutmise asemel saadetakse uusi mutatsioone ning need võivad sisaldada asünkroonseid toiminguid erinevalt

mutatsioonidest. Tegevusi kutsutakse välja kujul *store.dispatch(<tegevuse nimetus>, <tegevusega kaasa minev parameeter>)*. Moodulid võivad rakenduse oleku jagada mitmeks osaks, mis sisaldavad oma olekut, saajaid, mutatsioone, tegevusi ja mooduleid [59].

PlanMeet rakenduses on olekus erinevad andmebaasist pärinevad objektid ja nimekirjad objektidest, mida rakenduse vaates kasutavad. Mutatsioonides muudetakse olekut ja tegevustes tehakse päringuid serveripoolse rakendusega suhtlevatele teenustele ning saadud tulemuste põhjal uuendatakse mutatsioonide kaudu olekut. Saajaid ja mooduleid PlanMeet rakenduses ei kasutata.

4.2.5 Kasutajate ja kasutajagruppide funktsionaalsus

Kliendipoolsest rakendusest tehakse kasutaja sisse logimisel päring serveripoolsesse rakendusse. Sisse logimiseks küsitakse kasutajalt kasutajanime ja parooli. Serveripoolselt rakenduselt saadakse korrektsete andmete korral vastuseks kasutaja info ja JWT token. Kasutaja info ja JWT token salvestatakse sessiooni mällu, mis tähendab, et veebilehitsejas vahelehe sulgemisel mälu tühjendatakse. Kõigi vaadete kuvamisel kontrollitakse JWT tokeni olemasolu ning saadetakse see ka serveripoolsesse rakendusse päringute tegemisel kaasa.

4.2.6 Vaadete loomine

Varsemalt loodud kasutajakogemuse ja kasutajaliidese disainile põhinedes loodi rakenduse vaated. Vaated on jaotatud administraatori ja tavakasutaja vaadeteks ning need omakorda objektide nimekirja, muutmise ja lisamise vaadeteks. Administraator näeb kasutajate, organisatsioonide ja tähtpäevade nimekirja, lisamise ja muutmise vaateid. Tavakasutaja näeb kalendri ehk kohtumiste nimekirja, muutmise ja lisamise, küsitluste nimekirja, muutmise, lisamise ja vastamise ning TalTechi ruumide otsingu ja broneerimise vaateid. Vaadete sisu on tavakasutajatel piiratud nendega seotud info nägemisega. Vaadete .vue failid sisaldavad endas nii HTML-koodi kui ka TypeScript koodi. TypeScript kood on *<script></script>* märgiste vahel. Kogu HTML sisu peab olema märgiste <template></template> vahel. TypeScript koodis on vaate klass, mis laiendab liidest Vue ning on märgistatud annotatsiooniga *@Component*. Klass sisaldab kõiki vaate jaoks vajalikke funktsioone, seal hulgas elutsükli meetodid, nuppudele vajutamisest välja kutsutud meetodeid ja Vuex hoidlasse mutatsioonide saatmise ning

andmete pärimise meetodeid. Kui vaatel on parameetrid, siis need on tähistatud *@Prop()* annotatsiooniga.

Vaadete vahel liikumise kasutatakse Vue Router-it, mis on Vue ametlik ruuter. Selle abil seotakse rakenduse komponendid vastavate aadressidega ning komponentides on võimalik erinevatel viisidel nagu nime või aadressi järgi järgmisele lehele navigeerida. Kaasa saab anda ka parameetreid [69].

5 Hinnang loodud veebirakendusele

Lõputöö raames loodud rakendust võib hinnata oodatud ja saavutatud funktsionaalsuse põhjal. Hinnangu võib anda ka paindlikkusele võimalike edasiarenduste osas.

Arendusprotsessi jooksul töö autor laiendas oma teadmisi kaasaegsete veebirakenduste arendamise tehnoloogiate vallas. Analüüsiti erinevaid võimalusi ja õpiti kasutama sobivaimaid. Samuti kogeti ühe tervikliku rakenduse arendusprotsessi alguses lõpuni. Juurde õpiti viise, kuidas on võimalik teha päringuid eksisteerivasse lahendusse – TalTechi õppeinfosüsteemi, kus soovitud tegevused nõuavad sisse logimist. Samuti arendati teadmisi kasutajakogemuse ja kasutajaliidese disaini vallas.

5.1 Saavutatud funktsionaalsus

Lisas 2 kasutajalugudena kirjeldatud nõuded on enamuses valminud rakenduses täidetud. Rakenduse kalendrivaade, kohtumiste ja küsitluste lisamine on täielikult kasutatavad. TalTechi ruumide otsing ja broneerimine töötavad koodi kirjutatud ÕIS-i kasutajanime ja parooli abil. Laiemaks kasutamiseks tuleks lisada rakenduse töövoogu ÕIS-i sisse logimine.

Rakendus on jaotatud rollide põhjal – tavakasutaja ja administraatori vaated. Tavakasutaja näeb kalendrivaadet, kus on võimalik näha aktiivseid kohtumisi, neid lisada, muuta ja kustutada. Samuti näeb kalendrivaates aktiivseid tähtpäevi ja kasutajate sünnipäevi. Kohtumise loomisel on võimalik lisada kas konkreetne ajavahemik või luua küsitlus ühise aja leidmiseks. Küsitluse loomisel saab määrata mitmeid sobivaid kellaaegu ja kuupäevi. Samuti saab küsitluse loomisel muuta võimalikke vastusevariante. Kohtumise loomisel on võimalik määrata kindel asukoht või otsida sobivat asukohta TalTechi ruumide seast. Sobiva ruumi leidmisel on võimalik see ka broneerida. Kohtumisele saab kutsuda kas kõikide organisatsioonide liikmed, kindlate organisatsioonide liikmed või kindlad inimesed. Kasutajad näevad oma kutseid küsitluste vaates, kus nad saavad anda teada oma osalemisest või mitte osalemisest kindla aja korral või valida sobivad ajad mitme seast.

Administraator näeb infot kasutajate, tähtpäevade ja organisatsioonide kohta. Kasutajaid lisab administraator, kes eelnevalt kontrollib kasutaja kuulumist organisatsiooni või organisatsioonidesse. Administraator saab vajadusel kasutajate infot muuta ja kasutajaid eemaldada, kui nende kuuluvus organisatsioonidesse muutub. Lisaks saab administraator lisada, muuta ja eemaldada organisatsioone ning tähtpäevi. Tähtpäevade puhul saab määrata, milliste organisatsioonide liikmete jaoks need nähtavad on. Tähtpäevadena näidatakse kalendris ka liikmete sünnipäevi.

Esialgsetest nõuetest jäid hetkel välja erinevate kohtumiste kategooriate värvide muutmine ja kohtumiste filtreerimine kalendris. Lisaks kohtumiste perioodiline korraldamine ja mitmepäevaste kohtumiste lisamine. Veel tuleks lisada e-maili integratsioon, et liikmetele teateid saata ning kohtumisi automaatselt Google-i või Outlooki kalendrirakendustesse lisada. Eeltoodud nõuete täitmisega jätkub töö. Lisas 3 on toodud valminud serveripoolse ja kliendipoolse rakenduse avalikud lingid Bitbucket keskkonnas, mis võimaldavad ka teistel soovijatel koodiga tutvuda ja rakenduse avaldamisele kaasa aidata. Koodi kirjeldavas failis Bitbucketis on ka juhend rakenduse lokaalseks käivitamiseks.

5.2 Võimalikud edasiarendused

Loodud rakendus on üheks mitmetest TalTechi tudengiorganisatsioonidele mõeldud rakendustest. Tudengielu TalTechis on aga niivõrd suur ja aktiivne osa ülikoolikogemusest, et tudengiorganisatsioonide tegevust võiks tehniliste lahendustega veelgi enam toetada. Üks võimalustest oleks luua kõiki tudengiorgnisatsioonidele vajalikke tegevusi koondav portaal. Juba eksisteerib näiteks rahastuse taotlemiseks mõeldud veebiplatvorm. Teine variant oleks lisada olemasolevasse õppeinfosüsteemi eraldi ala tudengiorganisatsioonidele vajaliku jaoks nagu rahastus ja kohtumiste planeerimine. Veel üks võimalus oleks lisada kohtumiste planeerimise alajaotus eksisteerivasse TalTechi mobiilirakendusse või TalTechi tudengiportaali. Samas tuleks kuidagi tagada ligipääs ka ülikooli välistele tudengiorganisatsioonide liikmetele, mis hetkel kõigis eeltoodud valmis lahendustes puudub.

PlanMeet esmased edasiarendused on seotud veebirakenduse valmis saamisega avalikuks kasutamiseks, millest mitmed on toodud eelnevas peatükis **5.1 Saavutatud funktsionaalsus**. Veel võiks rakendusse lisada võimaluse näha, muuta ja kustutada

saadetud ÕIS-i ruumide broneerimise taotlusi ning saada teateid, kui taotlus kinnitatakse. Lisaks puudub hetkel vaade kasutaja seadete ja kasutajaandmete muutmiseks, mille peaks samuti enne avalikku kasutamist lisama. Seal oleks võimalik seadistada näiteks e-maili teadete saatmist, et soovijad saaksid teadetest keelduda. Kõigile vaadetele tuleks lisada ka lühike kasutusjuhend, et kõik tegevused rakenduses kiiresti selgeks saaksid.

Edasiarendusena oleks võimalik laiendada broneeritavate ruumide hulka ÕIS-ist kaugemale. Tudengiorganisatsioonid kasutavad oma kohtumiste korraldamiseks ka näiteks Mektory-t, tudengimaja ruume ja TalTechi saunu. Mektory ja TalTechi saunade jaoks oleks vaja lisada ka maksmise võimalus, sest nende kasutamine on tasuline. PlanMeet rakenduse üks võimalik suurem edasiarendus on mitmekeelsuse lisamine. Antud lõputöö skoobis oli hetkel vaid eesti keele kasutamine, kuid rakenduse laiemaks kasutamiseks oleks kindlasti vajalik ka inglise ja vene keele lisamine. Teine võimalik suurem edasiarendus oleks mobiilirakendus või veebirakendusele eraldi mobiilivaadete loomine. Tudengite üheks peamiseks suhtlusvahendiks on nutitelefon ning mugavus mobiilis on äärmiselt oluline.

Kui rakendus kasutusele võetaks, siis saaks teha edasisi uuringuid tudengiorganisatsioonide liikmete seas võimalike probleemide ja arenguvõimaluste leidmiseks ning seeläbi veelgi enam tudengiorganisatsioonide tegevust toetada.

6 Kokkuvõte

Bakalaureusetöö käigus loodi PlanMeet rakendus. Rakendus on mõeldud TalTechi tudengiorganisatsioonidele ühiste aegade leidmiseks ja ruumide broneerimiseks. Koosviibimisi saab korraldada nii organisatsioonisiseselt, organisatsioonide üleselt, organisatsioonide väliselt kui ka väiksemate gruppidega. Rakenduse üheks osaks on kalendrivaade, kus näeb oma väiksemate gruppide üritusi, organisatsioonisiseseid üritusi ja erinevate organisatsioonide ühiseid üritusi. Samuti on kalendrisse võimalik lisada erinevaid tähtpäevi, näiteks organisatsioonide sünnipäevad, organisatsiooni liikmete sünnipäevad ja üliõpilaskonna sünnipäev. Rakenduse teiseks osaks on küsitlused, et selgitada välja millal liikmed saaksid koosviibimistel osaleda. Küsitluste osana saab näha ka sobivate ruumide vabu aegu ning ruume broneerida. Ruumide valimiseks on võimalik määrata parameetreid nagu inimeste arv ja asukoht. Ruumide vabad ajad saadakse integratsioonist ÕIS-iga.

Rakendusega on saavutatud töö alguses püstitatud eesmärk. Esiteks, tudengiorganisatsioonidel on võimalik oma kohtumisi planeerida läbi ühe rakenduse. Kohtumiste planeerimine on lihtsam ja kiirem ning parandab organisatsioonide ülest ürituste korraldamist. Teiseks, TalTechi ruumide broneerimine on mugavam. Ruumide leidmiseks saab kasutada filtreerivaid parameetreid ning liikmete ühised ajad saab leida paralleelselt ruumide vabade aegadega.

Kasutatud kirjandus

[1] TalTech, „TalTech,“ [Võrgumaterjal]. Available: https://taltech.ee. [Kasutatud 21 03 2021].

[2] J. Pot, „Zapier,“ 18 11 2020. [Võrgumaterjal]. Available: https://zapier.com/blog/best-calendar-apps/. [Kasutatud 23 03 2021].

[3] „Doodle,“ [Võrgumaterjal]. Available: https://doodle.com. [Kasutatud 20 03 2021].

[4] T. Nechita, „Windows Report,“ 21 10 2019. [Võrgumaterjal]. Available: https://windowsreport.com/evolution-of-outlook/. [Kasutatud 23 03 2021].

[5] Google, „Google Blog,“ 07 07 2009. [Võrgumaterjal]. Available: https://googleblog.blogspot.com/2009/07/google-apps-is-out-of-beta-yes-really.html. [Kasutatud 23 03 2021].

[6] R. Woodgate, „How-To-Geek,“ 07 08 2019. [Võrgumaterjal]. Available: https://www.howtogeek.com/435975/how-to-show-an-outlook-calendar-in-google-calendar/. [Kasutatud 23 03 2021].

[7] „Microsoft Support,“ [Võrgumaterjal]. Available: https://support.microsoft.com/en-us/office/introduction-to-the-outlook-calendar-d94c5203-77c7-48ec-90a5-2e2bc10bd6f8. [Kasutatud 23 03 2021].

[8] „Google Help,“ [Võrgumaterjal]. Available: https://support.google.com/a/users/answer/9302892?hl=en. [Kasutatud 23 03 2021].

[9] „Calendly,“ [Võrgumaterjal]. Available: https://calendly.com/. [Kasutatud 24 03 2021].

[10] Woven, „The Ultimate Guide to Calendly,“ [Võrgumaterjal]. Available: https://woven.com/guides/calendly. [Kasutatud 24 03 2021].

[11] „Plandok,“ [Võrgumaterjal]. Available: https://plandok.com/et/. [Kasutatud 24 03 2021].

[12] Microsoft Docs, „Microsoft Bookings,“ 18 09 2020. [Võrgumaterjal]. Available: https://docs.microsoft.com/en-us/microsoft-365/bookings/bookings-overview?view=o365-worldwide. [Kasutatud 24 03 2021].

[13] „Microsoft Bookings,“ [Võrgumaterjal]. Available: https://www.microsoft.com/en-us/microsoft-365/business/scheduling-and-booking-app. [Kasutatud 24 03 2021].

[14] M. Hornostaiev, „Java, Python, and PHP: Which is Better for Server Backends?,“ Erminesoft, [Võrgumaterjal]. Available: https://erminesoft.com/java-python-and-php-which-is-better-for-server-backends/. [Kasutatud 30 03 2021].

[15] A. Goel, „Best Programming Languages to Learn in 2021 (for Job & Future),“ hackr.io, 21 01 2021. [Võrgumaterjal]. Available: https://hackr.io/blog/best-programming-languages-to-learn-2021-jobs-future. [Kasutatud 30 03 2021].

[16] H. Shnaider, „Top 7 Languages for Web App Development,“ FortySeven, 30 06 2020. [Võrgumaterjal]. Available: https://fortyseven47.com/news/top-7-languages-for-web-app-development/. [Kasutatud 30 03 2021].

[17] Maruti Techlabs, „How to Choose the Perfect Back-End Technology?,“ Maruti Techlabs, 26 07 2016. [Võrgumaterjal]. Available: https://marutitech.medium.com/how-to-choose-the-perfect-back-end-technology-5674db9c0192. [Kasutatud 30 03 2021].

[18] EBS Integrator, „Top 10 server-side programming languages in 2020,“ EBS Integrator, 27 07 2020. [Võrgumaterjal]. Available: https://ebs-integrator.com/blog/basics-top-10-server-side-programming-languages-in-2020/. [Kasutatud 30 03 2021].

[19] R. Shankar, „10 Best Java Frameworks to Use in 2021,“ hackr.io, 23 02 2021. [Võrgumaterjal]. Available: https://hackr.io/blog/java-frameworks. [Kasutatud 30 03 2021].

[20] V. Power, „Most Popular Java Web Frameworks in 2021,“ Rollbar, 23 07 2019. [Võrgumaterjal]. Available: https://rollbar.com/blog/most-popular-java-web-frameworks/. [Kasutatud 30 03 2021].

[21] Baeldung, „A Comparison Between Spring and Spring Boot,“ Baeldung, 24 03 2021. [Võrgumaterjal]. Available: https://www.baeldung.com/spring-vs-spring-boot. [Kasutatud 30 03 2021].

[22] GeeksForGeeks, „Difference between Spring MVC and Spring Boot,“ GeeksForGeeks, 23 11 2020. [Võrgumaterjal]. Available: https://www.geeksforgeeks.org/difference-between-spring-mvc-and-spring-boot/. [Kasutatud 03 04 2021].

[23] TutorialsPoint, „Hibernate - ORM Overview,“ TutorialsPoint, [Võrgumaterjal]. Available: https://www.tutorialspoint.com/hibernate/orm_overview.htm. [Kasutatud 03 04 2021].

[24] M. Tyson, „What is JPA? Introduction to the Java Persistence API,“ InfoWorld, 02 04 2019. [Võrgumaterjal]. Available: https://www.infoworld.com/article/3379043/what-is-jpa-introduction-to-the-java-persistence-api.html. [Kasutatud 03 04 2021].

[25] TutorialsPoint, „Spring - JDBC Framework Overview,“ TutorialsPoint, [Võrgumaterjal]. Available: https://www.tutorialspoint.com/spring/spring_jdbc_framework.htm. [Kasutatud 03 04 2021].

[26] O. Okinskas, „What is a package manager and why should you use one?,“ IDR Solutions, 11 07 2018. [Võrgumaterjal]. Available: https://blog.idrsolutions.com/2018/07/what-is-a-package-manager-and-why-should-you-use-one/. [Kasutatud 30 03 2021].

[27] Gradle Build Tool, „Gradle vs Maven Comparison,“ Gradle Build Tool, [Võrgumaterjal]. Available: https://gradle.org/maven-vs-gradle/. [Kasutatud 31 03 2021].

[28] JavaTpoint, „Gradle vs. Maven,“ JavaTpoint, [Võrgumaterjal]. Available: https://www.javatpoint.com/gradle-vs-maven. [Kasutatud 03 04 2021].

[29] N. Fatima, „A Quick Overview of Different Types of Databases,“ Astera, 28 03 2021. [Võrgumaterjal]. Available: https://www.astera.com/type/blog/a-quick-overview-of-different-types-of-databases/. [Kasutatud 03 04 2021].

[30] javaTpoint, „Types of Databases,“ javaTpoint, [Võrgumaterjal]. Available: https://www.javatpoint.com/types-of-databases. [Kasutatud 03 04 2021].

[31] C. Arsenault, „The Pros and Cons of 8 Popular Databases,“ KeyCDN, 20 04 2017. [Võrgumaterjal]. Available: https://www.keycdn.com/blog/popular-databases. [Kasutatud 03 04 2021].

[32] Guru99, „13 BEST Free Database Software (SQL Databases List) in 2021,“ Guru99, [Võrgumaterjal]. Available: https://www.guru99.com/free-database-software.html. [Kasutatud 03 04 2021].

[33] Educba, „Differences Between TypeScript vs JavaScript,“ Educba, [Võrgumaterjal]. Available: https://www.educba.com/typescript-vs-javascript/. [Kasutatud 14 04 2021].

[34] Html, „What is HTML?,“ Html, [Võrgumaterjal]. Available: https://html.com/. [Kasutatud 04 04 2021].

[35] W3C, „WHAT IS CSS?,“ [Võrgumaterjal]. Available: https://www.w3.org/Style/CSS/Overview.en.html. [Kasutatud 04 04 2021].

[36] A. Chernyakov, „Best Practices: Frontend Frameworks For Your Web Product,“ UpTech, [Võrgumaterjal]. Available: https://uptech.team/blog/frontend-frameworks-for-web-product. [Kasutatud 04 04 2021].

[37] BootstrapVue, „BootstrapVue,“ BootstrapVue, [Võrgumaterjal]. Available: https://bootstrap-vue.org/. [Kasutatud 14 04 2021].

[38] Bootstrap, „Build fast, responsive sites with Bootstrap,“ Bootstrap, [Võrgumaterjal]. Available: https://getbootstrap.com/. [Kasutatud 04 04 2021].

[39] J. v. Gumster, „6 places to host your git repository,“ OpenSource, 30 08 2018. [Võrgumaterjal]. Available: https://opensource.com/article/18/8/github-alternatives. [Kasutatud 03 04 2021].

[40] GitLab, „What is GitLab?,“ GitLab, [Võrgumaterjal]. Available: https://about.gitlab.com/what-is-gitlab/. [Kasutatud 03 04 2021].

[41] GitHub, „GitHub,“ GitHub, [Võrgumaterjal]. Available: https://github.com/. [Kasutatud 03 04 2021].

[42] BitBucket, „A brief overview of Bitbucket,“ BitBucket, [Võrgumaterjal]. Available: https://www.atlassian.com/software/bitbucket/guides/getting-started/overview. [Kasutatud 03 04 2021].

[43] V. Singh, „Best Java IDE 2021 | Most Popular Java IDE,“ hackr.io, 23 01 2021. [Võrgumaterjal]. Available: https://hackr.io/blog/best-java-ides. [Kasutatud 03 04 2021].

[44] M. Heller, „Choosing your Java IDE,“ InfoWorld, 18 12 2018. [Võrgumaterjal]. Available: https://www.infoworld.com/article/3114167/choosing-your-java-ide.html. [Kasutatud 03 04 2021].

[45] Slant, „What are the best IDEs for web development?,“ Slant, 24 03 2021. [Võrgumaterjal]. Available: https://www.slant.co/topics/8150/~ides-for-web-development. [Kasutatud 04 04 2021].

[46] Visual Studio Code, „Visual Studio Code,“ Visual Studio Code, [Võrgumaterjal]. Available: https://code.visualstudio.com/. [Kasutatud 04 04 2021].

[47] Geekflare Editorial, „11 Awesome User Flow Tools for UX Design,“ Geekflare, 23 01 2020. [Võrgumaterjal]. Available: https://geekflare.com/best-ux-user-flow-tools/. [Kasutatud 15 04 2021].

[48] UXPin, „UXPin,“ UXPin, [Võrgumaterjal]. Available: https://www.uxpin.com/. [Kasutatud 16 04 2021].

[49] Spring Initializr, „Spring Initializr,“ Spring Initializr, [Võrgumaterjal]. Available: https://start.spring.io/. [Kasutatud 05 04 2021].

[50] Spring, „Spring Boot,“ Spring, [Võrgumaterjal]. Available: https://spring.io/projects/spring-boot#learn. [Kasutatud 05 04 2021].

[51] StackChief, „Which Version of Java Should You Use?,“ StackChief, 16 02 2021. [Võrgumaterjal]. Available: https://www.stackchief.com/blog/Which%20Version%20of%20Java%20Should%20You%20Use%3F. [Kasutatud 05 04 2021].

[52] Baeldung, „Introduction to Project Lombok,“ Baeldung, 09 12 2020. [Võrgumaterjal]. Available: https://www.baeldung.com/intro-to-project-lombok. [Kasutatud 05 04 2021].

[53] Spring, „Accessing Data with JPA,“ Spring, [Võrgumaterjal]. Available: https://spring.io/guides/gs/accessing-data-jpa/. [Kasutatud 09 04 2021].

[54] HeidiSQL, „What's this?,“ HeidiSQL, [Võrgumaterjal]. Available: https://www.heidisql.com/. [Kasutatud 09 04 2021].

[55] bezkoder, „Spring Boot + Vue.js: Authentication with JWT & Spring Security Example,“ 29 11 2020. [Võrgumaterjal]. Available: https://bezkoder.com/spring-boot-vue-js-authentication-jwt-spring-security/. [Kasutatud 15 04 2021].

[56] A. Bouchefra, „A Beginner’s Guide to Vue CLI,“ SitePoint, 25 06 2019. [Võrgumaterjal]. Available: https://www.sitepoint.com/vue-cli-intro/. [Kasutatud 15 04 2021].

[57] Babel, „Babel,“ Babel, [Võrgumaterjal]. Available: https://babeljs.io/. [Kasutatud 15 04 2021].

[58] Vue Router, „Vue Router,“ Vue Router, [Võrgumaterjal]. Available: https://router.vuejs.org/. [Kasutatud 15 04 2021].

[59] Vuex, „What is Vuex?,“ Vuex, [Võrgumaterjal]. Available: https://vuex.vuejs.org/. [Kasutatud 15 04 2021].

[60] Vue Loader, „Linting,“ Vue Loader, [Võrgumaterjal]. Available: https://vue-loader.vuejs.org/guide/linting.html#eslint. [Kasutatud 15 04 2021].

[61] MadeWithVueJs, „Vue 3 – A roundup of infos about the new version of Vue.js,“ MadeWithVueJs, 31 03 2021. [Võrgumaterjal]. Available: https://madewithvuejs.com/blog/vue-3-roundup. [Kasutatud 15 04 2021].

[62] TypeScript, „What is a tsconfig.json,“ TypeScript, [Võrgumaterjal]. Available: https://www.typescriptlang.org/docs/handbook/tsconfig-json.html. [Kasutatud 15 04 2021].

[63] Make a README, „README 101,“ Make a README, [Võrgumaterjal]. Available: https://www.makeareadme.com/. [Kasutatud 15 04 2021].

[64] Node Js, „What is the file `package.json`?,“ Node Js, 26 08 2011. [Võrgumaterjal]. Available: https://nodejs.org/en/knowledge/getting-started/npm/what-is-the-file-package-json/. [Kasutatud 15 04 2021].

[65] Node Js, „The package-lock.json file,“ Node Js, [Võrgumaterjal]. Available: https://nodejs.dev/learn/the-package-lock-json-file. [Kasutatud 15 04 2021].

[66] Bitbucket, „.gitignore,“ Bitbucket, [Võrgumaterjal]. Available: https://www.atlassian.com/git/tutorials/saving-changes/gitignore. [Kasutatud 15 04 2021].

[67] Microsoft Docs, „Create portable, custom editor settings with EditorConfig,“ Microsoft Docs, 09 02 2020. [Võrgumaterjal]. Available: https://docs.microsoft.com/en-us/visualstudio/ide/create-portable-custom-editor-options?view=vs-2019. [Kasutatud 15 04 2021].

[68] browserslist, „browserslist,“ 25 02 2021. [Võrgumaterjal]. Available: https://github.com/browserslist/browserslist. [Kasutatud 15 04 2021].

[69] Vue Router, „Vue Router,“ Vue Router, [Võrgumaterjal]. Available: https://router.vuejs.org/. [Kasutatud 16 04 2021].

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

Mina, Liisi Lota Pals

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose „TalTechi tudengiorganisatsioonide kohtumiste planeerimise rakendus PlanMeet“, mille juhendaja on Meelis Antoi
   1.1. reprodutseerimiseks lõputöö säilitamise ja elektroonse avaldamise eesmärgil, sh Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
   1.2. üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. Olen teadlik, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autorile.
3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

17.05.2021

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

Lisa 2 – Nõudeid kirjeldavad kasutajalood

Lugu 1: Tudengiorganisatsiooni liikmena tahan näha kalendrit, mis sisaldaks kõiki minuga seotud kohtumisi ja tähtpäevi, et omada ülevaadet oma ajast.

- K1: Liikmena tahan näha oma väiksemate gruppide, oma organisatsiooni siseseid ja kõikide organisatsioonide vahelisi kohtumisi, et omada ülevaadet oma ajast.
- K2: Liikmena tahan näha oma organisatsiooni, teiste organisatsioonide avalikke, TalTechi ja Eesti riiklikke tähtpäevi, et tähtsaid päevi meeles pidada.
- K3: Liikmena tahan, et kohtumised ja tähtpäevad oleksid eristatud värvidega, et saaksin üritusi kergesti eristada.
- K4: Liikmena tahan filtreerida kohtumisi ja tähtpäevi vastavalt oma soovile, et leiaksin otsitava mugavalt ja kiirelt.
- K5: Liikmena tahan kalendris näha kohtumiste pealkirju ja toimumisaegu, et saada kiiret ülevaadet oma plaanidest.

Lugu 2: Kohtumise korraldajana tahan luua küsitlusi oma kaaslastele, et ühiseid kohtumiste aegu leida.

- K1: Korraldajana tahan määrata kõik võimalikud kuupäevad ja kellaajad kohtumise toimumiseks, et osalejad saaksid nende vahel valida.
- K2: Korraldajana tahan võimalikke aegu automaatselt genereerida, kui võimalikud ajad korduvad perioodiliselt.

Lugu 3: Kohtumise korraldajana tahan luua uusi kohtumisi, et kaaslastega kokku saada.

- K1: Korraldajana tahan määrata kohtumise pealkirja, kirjelduse, alguse ja lõpu aja ning asukoha, et anda infot osalejatele.
- K2: Korraldajana tahan kutsuda kohtumisele kindlaid inimesi, terve organisatsiooni, mitmeid organisatsioone või kõik organisatsioonid, et kohtuda teemasse puutuvate inimestega.

Lugu 4: Kohtumise korraldajana tahan otsida TalTechi ruume, näha nende vabu aegu ning broneerida, et leida kohtumiseks sobiv ruum.

- K1: Korraldajana tahan otsida ruume hoonete, korrusete, ruumi numbrite, mahutavuse ja/või varustuse järgi, et leida kõik sobivad ruumid kohtumiseks.
- K2: Korraldajana tahan otsida ühte kindlat aega või mitut aega korraga, et leida kõik võimalused kohtumise korraldamiseks.
- K3: Korraldajana tahan ruume korduvalt broneerida, kui kohtumised toimuvad perioodiliselt.
- K4: Korraldajana tahan saata taotluse ruumi broneerimiseks, kui sobiv ruum on leitud.

Lugu 5: Ürituse korraldajana tahan hallata enda loodud kohtumisi ja nende osalejaid.

- K1: Korraldajana tahan oma kohtumisi kustutada, kui nende toimumine pole võimalik.
- K2: Korraldajana tahan oma kohtumisi muuta, kui nende aeg või toimumiskoht muutuvad.
- K3: Korraldajana tahan kutsuda kohtumistele uusi kasutajaid, kui teemasse puutuvate inimeste hulk suureneb.

Lugu 6: Kohtumisel osalejana tahan vastata kohtumisel osalemise kutsele, et teavitada korraldajat oma otsusest.

- K1: Osalejana tahan valida vastusevariantide „Jah“, „Ei“ ja „Võib-olla“ vahel, et määrata oma osalus kohtumisel.
- K2: Osalejana tahan saada teavituse e-mailile, kui mulle on saabunud uus kutse kohtumisele.
- K3: Osalejana tahan muuta oma osalust kohtumisel, kui minu plaanid muutuvad.

Lugu 7: Kohtumisel osalejana tahan vastata küsitlusele kohtumise toimumisaja kohta, et kohtumine toimuks mulle sobival ajal.

- K1: Osalejana tahan valida ühe või mitu sobivat kuupäeva või kellaaega valikus olevatest võimalustest, et teavitada korraldajat kõigist mulle sobivatest aegadest.

- K2: Osalejana tahan valida vastusevariantide „Jah“, „Ei“ ja „Võib-olla“ vahel, et määrata aja sobivust mulle.

- K3: Osalejana tahan saada teavituse koos kalendrilingiga e-mailile, kui lõplik aeg ürituse toimumiseks on valitud.

Lugu 8: Kohtumisel osalejana tahan näha kohtumise infot ning saada teavitusi muutustest, et olla õigel ajal õiges kohas.

- K1: Kohtumisel osalejana tahan saada teavitusi koos kalendrilingiga e-mailile, kui kohtumise aeg või koht muutuvad, et saaksin ka oma plaanides muudatusi teha.

- K2: Osalejana tahan näha kohtumise pealkirja, toimumisaega, asukohta, korraldajat ja osalejaid, et teada kohtumise infot.

Lugu 9: Rakenduse administraatorina tahan hallata kasutajaid ja nende õiguseid.

- K1: Administraatorina tahan lisada kasutajaid, et neil oleks võimalik näha kalendrit ning uusi kohtumisi lisada.

- K2: Administraatorina tahan kasutajaid eemaldada, kui nad pole enam organisatsioonide liikmed.

- K3: Administraatorina tahan kasutajate õigusi muuta, kui nad liituvad uute organisatsioonidega või lahkuvad mõnest organisatsioonist.

- K4: Administraatorina tahan lisada, muuta ja eemaldada organisatsiooni siseseid, organisatsiooni avalikke, TalTechi ja Eesti riiklikke tähtpäevi, kui need muutuvad.

Lisa 3 – Rakenduse lähtekood

Versioonihalduskeskkonna link serveripoolse rakenduse jaoks: https://bitbucket.org/lotaliisi/planmeet-backend.

Versioonihalduskeskkonna link kliendipoolse rakenduse jaoks: https://bitbucket.org/lotaliisi/planmeet-frontend.