TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Rasmus Riismann 193931IAIB

SÜNKRONISEERITUD MEEDIA ESITAMISE VEEBIRAKENDUS: WANDERFUL DAY

Bakalaureusetöö

Juhendaja: Gert Kanter
Doktorikraad

Tallinn 2022

Autorideklaratsioon

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

Autor: Rasmus Riismann

01.05.2022

Annotatsioon

Käesoleva lõputöö eesmärgiks oli luua veebirakenduse tagaliides ning algeline eesliides, seadistada veebiserver, suunata domeen sellele ning käivitada püsiv veebileht. Töö käigus valminud rakendus on esimene osa suuremast projektist.

Rakendusesiseselt on võimalik kasutajal viibida teiste kasutajatega kas avalikes või privaatsetes tubades, vestelda nendega läbi tekstiakna ning panna mängima kas YouTube lingilt saadud video või üleslaetud MP3 helifail. Juhul, kui kasutaja ühineb ruumiga, kus meedia juba käib, siis liituja kuuleb seda pooleliolevalt kohalt.

Rakendus on mõeldud inimestele, kes tahavad videoid või audiofaile reaalajas koos teiste inimestega vaadata või kuulata.

Lõputöö on kirjutatud eesti keeles keeles ning sisaldab teksti 17 leheküljel, 5 peatükki, 9 joonist, 0 tabelit.

Abstract

Web Application for Synchronized Media Presentation: Wanderful Day

The goal of this thesis was to create a backend and a basic frontend for a web application. Other goals include redirecting a domain to the server and creating a dedicated web service. The outcome of this project is the first piece of a larger project.

In the app, the user can create or join either a public or a private room with others. Users, who are in the same room, can chat with each other through a message window or request songs from YouTube with a link. Users can also play their own uploaded MP3 files. If a user joins a room that already has audio playing, it will be synchronized and start from where others are currently hearing it from.

This application is meant to be used by people who want to watch videos or listen to audio files together with other people.

The thesis is in Estonian and contains 17 pages of text, 5 chapters, 9 figures, 0 tables.

Lühendite ja mõistete sõnastik

3D	Kolme dimensiooniline
AJAX	*Asynchronous JavaScript and XML*, asünkroonne JavaScript ja XML
API	*Application Programming Interface*, rakendusliides
AWS	Amazon Web Services
CSS	*Cascading Style Sheets*, kaskaadlaadistik
EC2	*Elastic Compute Cloud*, elastne pilveandmetöötlus
HTML	*HyperText Markup Language*, hüpertekst-märgistuskeel
HTTP	*HyperText Transfer Protocol*, hüperteksti edastusprotokoll
HTTPS	*HyperText Transfer Protocol Secure*, hüperteksti turvaline edastusprotokoll
IP	*Internet Protocol*, interneti protokoll
JS	*JavaScript*
MB	*Megabyte*, megabait
NPM	*Node Package Manager*, NodeJs'i pakihaldur
PM2	Protsessihaldur
SSH	*Secure Shell*, turvakest
SSL	*Secure Sockets Layer*, transpordikihi turbeprotokoll
URL	*Uniform Resource Locator*, internetiaadress

Sisukord

Jooniste loetelu

1 Sissejuhatus

Tänapäeval on mitmeid erinevaid suhtluskanaleid, mitmete erinevate sihtrühmade ning otstarvetega. Leidub selliseid, kus oluliseks osaks peetakse sündmuste märkimist kalendrisse, ekraani jagamist või videopildi näitamist (Näiteks MS Teams [1]). Harva kohtab ka suhtluskanaleid, mille põhieesmärk on mängida ning sünkroniseerida heli kasutajate vahel (Näiteks Turntable Fm [2]). Kui osata otsida, siis võib leida isegi suhtluskanaleid, mis asuvad mängutaolistes keskkondades (Näiteks VR Chat [3]).

Nende analüüsimisest sündis idee panna osadest kokku mängutaoline ning interaktiivne rakendus, mille eesmärk on pakkuda ning imiteerida seltskonnaga automatka (n-ö *roadtrip*'i) kogemust, aga läbi interneti. Tekkis mõte ehitada platvorm, kus on võimalik inimestega tutvuda ning suhelda, muusikat kuulata ning nautida mööduvat ümbruskonda. Eesmärk on anda kasutajale kogemus, et ta tegelikult ka on sõiduvahendis ning veedab inimestega aega.

Kuna sellise täiemahulise platvormi ehitamiseks kulub tohutult aega ja inimresurssi, siis selle lõputöö eesmärgiks on ehitada prototüüp, mis demonstreerib kõige olulisemat põhifunktsionaalsust. Täpsemalt seatakse üles tagaliides ning algeline eesliides, mis annab ligipääsu tagaliidese funktsionaalsusele.

Lõputöös on täpsemalt välja toodud sammud, mida läbiti, et üles seada toimiv veebirakendus, mis võimaldab kasutajatel ruume (mis täieliku rakenduse kontekstis on sõiduvahend) luua, nendega liituda, tekstipõhist vestlust pidada ning kahel erineval viisil heli mängida.

2 Taust

Suhtluskanaleid ning nende sihtotstarbeid on palju. Selle töö raames käsitletakse suhtluskanalitüüpi, mille eesmärk on pakkuda keskkonda vaba aja veetmiseks. Käsitletavas peatükis arutatakse üldist ideed ja selle sündi, hetkel olemasolevaid sarnaseid lahendusi ning ka selle lõputöö eesmärki.

2.1 Idee tutvustus

Kui rääkida suurest pildist, siis lõputöö üldine eesmärk on pakkuda kasutajale automatka (ingl *roadtrip*) kogemust. Seda pakub 3D keskkonnas pidevalt muutuv keskkond ümber sõidutee, millel asetseb sõiduvahend, mis oleks ühtlasi ka rakenduse keskpunkt. Sõiduvahendisse mahuks vastavalt selle tüübile teatud arv kasutajaid, ning nad oleks esitatud ka mängusiseste tegelastena. Mängijatel oleks võimalik suhelda omavahel kas rääkides või kirjutades. Sõiduvahendis oleks võimalik muuta mängivat heli, mida kõik samaaegselt kuulevad, samalaadselt nagu päris elus. Mängijatele pakutakse erinevaid tegevusi, et neil oleks võimalik teiste mängijatega koos midagi teha.

2.2 Olemasolevad lahendused

Tänapäeval on tohutult palju suhtluskanaleid, kuid harva leiab selliseid, kus on kolm põhilist idees kirjeldatud põhipunkti: suhtlus, muusika ning muutuv ümbrus. Lõputöö autor on kasutanud mitu aastat suhtlusportaali Discord[4], mis katab ära kaks nendest punktidest, milleks on suhtlus ja muusika (mis on teostatud arvutiprogrammide poolt). Töö käigus avastas autor, et väga sarnase teemaga tegelevad VRChat'i [3] alammängud.

2.3 Lõputöö skoobi määramine

Kuna sellise rakenduse realiseerimine on suuremahuline ning vajaks mitmeliikmelist meeskonda, siis lõputöö raames realiseeritakse põhifunktsionaalsusega tagaliides ning algeline suhtluskanali eesliides, mis suudab päringuid saata ning võtta vastu tagaliideselt. Lisaks sellele seadistatakse üles veebiserver, mis võimaldab rakendusele ligipääsu läbi interneti. Iseseisva lõputööna valmistab Carl Eric Reinsberg projektile 3D keskkonnaga kasutajaliidese.

3 Teostus

Selles peatükis kirjeldatakse idee elluviimist. Arutatakse millised osad tahetakse lõputöö raames ära teha, milliseid tehnoloogiaid kasutatakse ning tuuakse välja ka arengufaasis kohatud iseärasused.

3.1 Tagaliidese nõuete analüüs

Lõputöö eesmärk on valmistada tagaliides ning algeline eesliidese osa, mis suhtleb tagaliidesega. Rakenduses peab olema toimiv ruumide süsteem, kus igas ruumis on avatud suhtluskanal. Ühes ruumis viibivad kasutajad peavad saama saata sõnumeid läbi sisendi ning kuvada teistelt saadud sõnumeid tekstiaknas. Kasutajad peaksid saama määrata endale nime (limiteeritud pikkusega), esialgse ühenduse korral määratakse kasutaja nimeks “Guest_<numbrid>“, kus numbrid on suvaline neljakohaline arv. Kõik kasutajad peavad saama ruume luua koos nimega ning otsustada, kas ruum on privaatne või mitte. Eesliideses peaks olema ruumiloendur, kus on näidatud kõik hetkel aktiivsed avalikud ruumid. Ruumil on nimi ja identifikaator, kus identifikaator on viiekohaline suvaliselt genereeritud tähejada. See võimaldab genereerida peaaegu 12 miljonit erineva identifikaatoriga ruumi.

Olles ruumis peaks igal kasutajal olema võimalus kõikidele ruumis viibijatele mängima panna meediat kahest erinevast allikast, YouTube [5] lingina või üles laadida enda arvutist MP3 fail, mis suuruselt ei ole üle 20MB. Juhul kui ruumis mängib helifail ning kasutaja ühineb antud ruumiga, siis kasutaja peaks kuulma faili tema praegusest

esitusajast, mitte algusest. Eesliideses peaks olema leitav helitugevuse regulaator. Eesliides üldiselt ei tohiks katta väga suurt osa ekraanist, sest tegu ei ole potentsiaalse üldrakenduse põhifookuses oleva elemendiga.

3.2 Kasutatud tehnoloogiad

Selles peatükis on välja toodud tähtsamad tehnoloogiad, kiire ülevaade nendest ning kirjutatud, kuidas ning mis osi nendest kasutati projektis. Kindlasti ei piirdu kasutatud tehnoloogiate nimekiri vaid järgnevates alapeatükkides väljatoodutega, kuid need mängisid võrdlemisi suurt rolli projekti vältel. Töö autoril on kokkupuudet olnud mõne järgneva tehnoloogiaga ning seetõttu osutusid need ka valituks.

3.2.1 Veebiserver – AWS EC2

Rakenduse ülalolekuaja maksimeerimiseks peab see jooksma serveri peal. Tänapäeval üks populaarseimatest valikutest on Amazon Web Services [6], mis pakub tellitavat pilveandmetöötluse platvormi. Lõputöös võeti kasutusele AWS'i poolt pakutav EC2 lahendus, kus käivitati „*t3.micro*" tüüpi virtuaalserver, millel on 1 GB muutmälu ning 8 GB kettamaht. Lõputöö raames on see piisavalt palju, et demonstreerida põhifunktsionaalsust, kuid kindlasti on vaja suurendada, kui tekib plaan projektiga edasi minna.

AWS juhtpaneel võimaldab portide sulgemist ning avamist valitud aadresside puhul. Rakenduse jaoks avati pordid 80 (HTTP port) ning 443 (HTTPS port) kõigile ligipääsetavaks. Port 22 avati, et oleks võimalik SSH'ga serverisse ühendada. Avati ka port 3000, mida PM2 kasutab rakendusesiseselt (hiljem täpsemalt).

Juhtpaneelist on võimalik ka valitud ajahetketel teha jooksvatest eksemplaridest tagavarakoopia, mis salvestab serveri seadistuse. Juhul kui midagi läheb valesti serveril, siis on võimalik taastada server töötavasse ajahetke tagasi. Lõputöö käigus tehti ka tagavara koopia, kuid õnneks seda kasutama ei pidanud.

3.2.2 Domeeni suunamine – Route 53

Rakendusele suunav domeen on *www.wanderful.day* või *wanderful.day* mis on ostetud NameCheap [7] domeeninimede registreerimisteenusest. Domeeni nimeserverina võeti kasutusele AWS poolt pakutud Route 53 teenus. See suunab kõik päringud, mis tehakse domeenile, juhtpaneelis seadistatud serverile suunavale elastsele IP aadressile. Serveriga ühenduse turvaliseks muutmiseks pidi looma SSL sertifikaadi, mis võimaldab HTTPS päringute saatmise. Seda tehti CertBot abiga serveri peal.

3.2.3 NodeJs

Tagaliideses kasutati Node.js [8] käitusteeki, sest see on kiire, asünkroonne ning sündmuspõhine, mis tähendab, et on väga hea valik kirjeldatud rakenduse loomiseks. Rakenduse kasutamisel tehakse palju päringuid, mis tuleb rahuldada asünkroonselt. NodeJs pakub ka NPM teenust, mis teeb moodulite haldamise väga mugavaks.

3.2.4 Nginx

Nginx [9] on vabavaraline tarkvara, mida kasutatakse arvutivõrkudes peamiselt veebiserverina, koormusjaoturina, meedia voogedastusabivahendina. Projektis kasutati Nginx'i peamiselt proksina, et suunata päringuid erinevate portide vahel. Kui päring tehakse HTTP port 80 või HTTPS port 443 pealt, siis suunatakse see edasi port 3000 peale, mida serveris jooksev fail kasutab. Samuti kasutati Nginx'i seadistus võimalusi, et määrata ära maksimaalne failisuurus, mida server on nõus ühest post-päringust vastu võtma (20 MB). Nginx võimaldab ka logifailide tekitamist kasutajapoolt täpsustatud informatsiooni salvestamiseks. Kui kasutaja ühendab serveriga, siis salvestatakse aeg, päringu lähteaadress ning millise veebibrauseriga päring tehti.

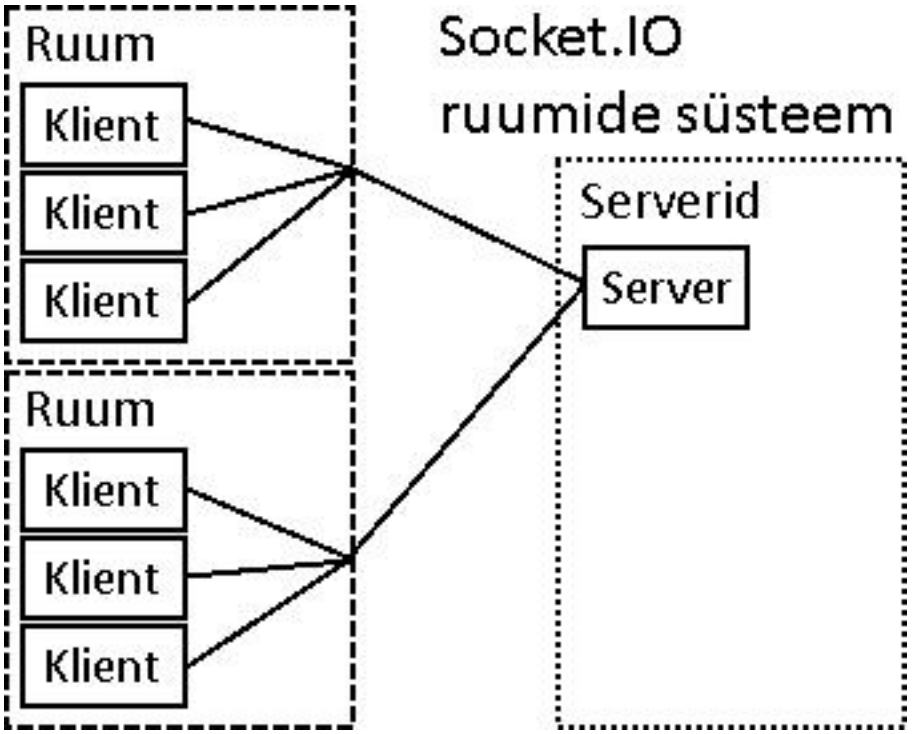
3.2.5 PM2

PM2 [10] on serveri taustal jooksev protsessihaldur, mis võimaldab programme jooksutada virtuaalselt maasolekuajata. Projektis kasutatakse PM2'te, et hoida serveril jooksvat programmi alati töös. PM2 annab ka ligipääsu konsoolile, et lugeda serveris välja prinditud veateateid. Et paremini aru saada, mida server teeb, kirjutati peaaegu iga päringu jaoks eraldi konsooliteade. See tekitas võimaluse reaalajas jälgida päringuid ning programmi töövoogu.

3.2.6 Socket.IO

Valdav enamik programmist on üles ehitatud Socket.IO [11] peale. Tegemist on teegiga, mis on ülesehitatud WebSocket protokollile. Socket.IO pakub vahendeid, et üles seada kahesuunaline sündmuspõhine suhtluskanal kasutajate vahel. Seda kanalit hoitakse lahti kuni kasutaja rakenduse kinni paneb.

Projekti raames kasutati Socket.IO ruumide süsteemi, et tekitada üksteisest eraldatud kasutajakogumid, ehk ruumid. Ühe serveriga saab ühendatud olla mitu ruumi. Joonisel (Joonis 1) on välja toodud näide võimalikust ühendussüsteemist, kus serveris on kaks ruumi aktiivsed ning mõlemas ruumis on kolm klienti sees. Laialdast kasutust leidis ka sündmustesüsteem, mida kasutas nii klient kui ka server ise, kus klient saatis sündmuseid serverile, ning sealt saadeti sündmuseid tagasi klientidele.

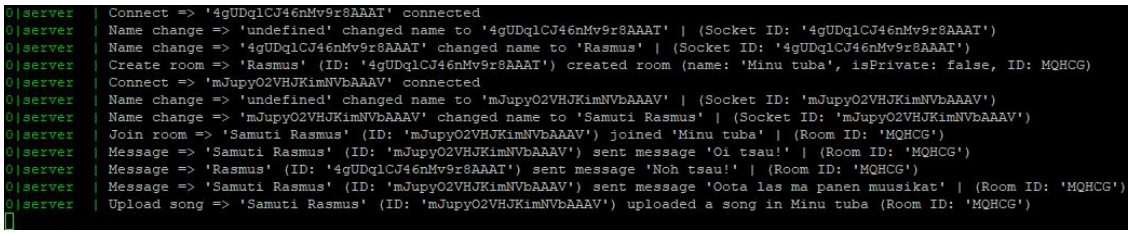


Joonis 1. Socket.IO ruumide süsteem

3.3 Tagaliides

Tagaliidese koodi peamised eesmärgid on lahti hoida suhtluskanaleid kasutajate vahel, talletada ruumides mängivate laulude informatsiooni ning edastada andmevoona kasutajate poolt üleslaetud helifaile. Nagu mainitud, siis programm kasutab Socket.IO sündmuspõhist andmevahetust. Server kuulab kasutajate poolt tulevaid päringuid/sündmusi ning teeb vastavaid muudatusi ning saadab välja päringuid. Kliendi ühendamisel serveriga saadab server kliendile failid, mida kasutajale kuvada.

Serveripoolsesse koodi sai kirjutatud ka konsooliteadete väljastamise peaaegu iga päringu puhul. See võimaldas, koos PM2 logimissüsteemiga, serveri peal reaalajas tuvastada ning leida vigu töövoos. Joonisel (Joonis 2) on näide võimalikest konsooliteadetest.



Joonis 2. Reaalajas teateid viskav konsool

3.4 Muusika ja heli

Selles peatükis arutatakse helifailide ja nende informatsiooni käsitlemist ning täpsustatakse tagaliidese nõudeid ning nende täitmisviise. Üks tagaliidese peamistest ülesannetest on helifailide informatsiooni haldamine, täpsemalt, mis ruumis mängitakse millist helifaili ning millise algusajaga. Selle tähtsus tuleneb heli sünkroniseerimisvajadusest. Kui hetkel ruumis mängib muusika ning keegi vahetab selle ära, siis vana heliallikas läheb kinni, mitte ei hakka lisakihina mängima.

3.4.1 Videod YouTube’st

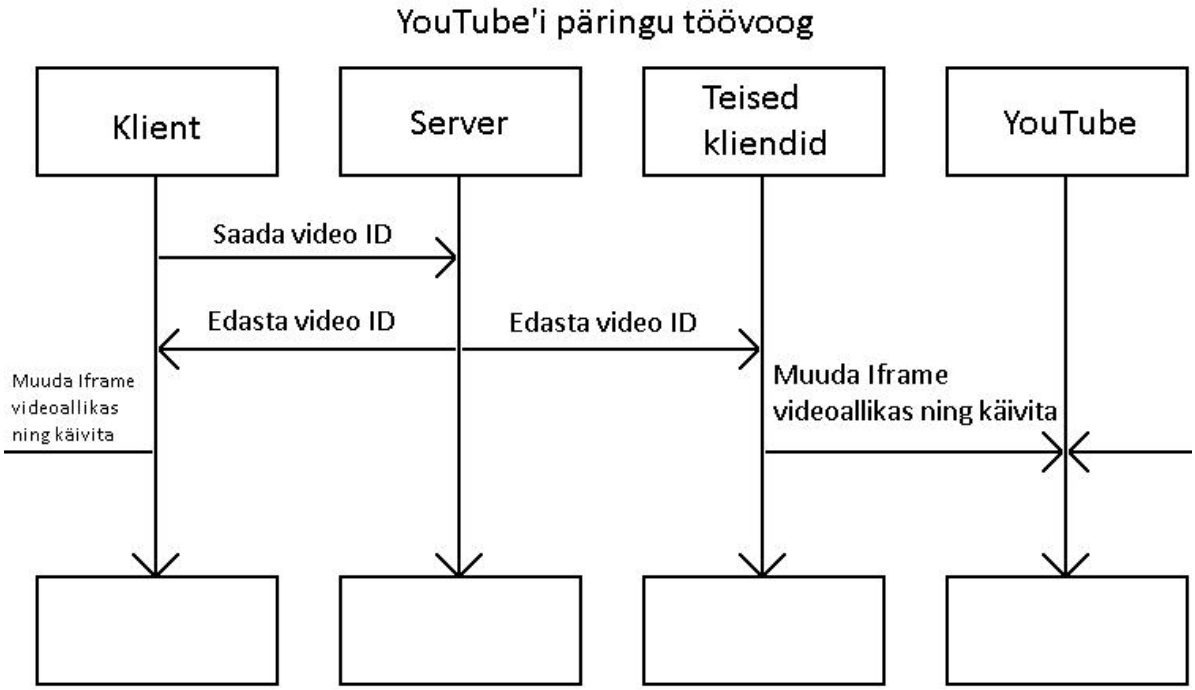
Arendusfaasis katsetati kolme erineva viisiga meedia mängimiseks läbi YouTube’i. Esialgu lisati leheküljele manustatud aken, mille saab peaaegu iga YouTube video alt, kui avada jagamisvalikud. Seejärel ehitati sõnumivahetussüsteemi sisse video mängimiseks mõeldud käsk *„!play <YouTube URL>“,* mis jagab lingi teistele hetkel ruumis viibivatele kasutajatele. Selle saavutamiseks kasutati Socket.IO sündmuseid,

käsu kirjutamisel väljastas klient sündmuse serveri suunas ning kui see kohale jõudis, siis server saatis sündmuse kõikidele klientidele, kes viibisid samas ruumis. Sündmuse saatmine saatis kaasa ka video URL'i, mille kliendipoolne JavaScript tööle pani uues manustatud YouTube aknas.

Seejärel uuriti ja katsetati ka YouTube'st ainult muusika edastamist läbi voogude. Mõte oli selles, et server teeb päringu video saamiseks andmevoona, aga ei esita ega salvesta seda, vaid eraldab audio ning saadab edasi kasutajatele. Selline protsess tegelikult töötas, aga otsustati seda mitte kasutada, sest see ei järgi YouTube'i poolseid teenustingimusi.

Seetõttu otsustati siiski video manustamise juurde tagasi minna. Leiti, et on olemas YouTube Iframe Player API [11], mis teeb manuste sisu ning käitumise muutmise võrdlemisi lihtsaks. Rakenduse viimases versioonis on YouTube manuse video vahetamiseks sama süsteem, mis esimeses, kus käsuga saab teistele ruumisviibivatele kasutajatele saata video URL'i. Joonisel (Joonis 3) on näidatud päringu liikumist erinevate programmide töövoos.

Hiljem, pärast tagasiside voore, eraldati YouTube videote esitamise funktsionaalsus sõnumivahetussüsteemist ning tekitati selle jaoks eraldi sisend.

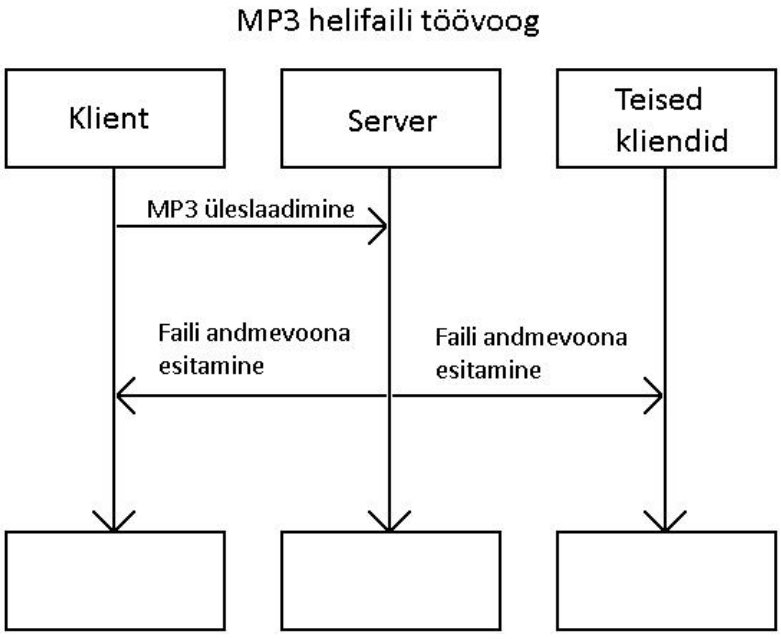


Joonis 3. Osapoolte töövood YouTube'i päringu korral

3.4.2 MP3 helifailid

Kasutajal on võimalus muusika jagamiseks serverisse üles laadida enda *mp3* faile, mille suurus ei ületa 20 MB. Valitud suurus võimaldab üles laadida kuni umbes 12 – 15 minuti pikkuseid laule ning kindlustab ka, et kettaruum ei täituks väga kiiresti. Faili üleslaadimist teeb kliendipoolne kood AJAX *post*-päringuga ning võetakse serveri poolt vastu Express raamistiku meetoditega. Serveris salvestatakse see kettale ja pannakse nimeks hetkeline ruumikood. Kuna ruumis saab korraga mängida ainult üks laul, siis nimekonflikti ei teki. Laulu üleslaadimisel saadab server välja sündmuse selle ruumi kasutajatele, et nad oleksid valmis vastu võtma muusikat andmevoona.

Üks puudujääk sellise meetodiga on rakenduse skoobi muutmine. Nagu mainitud on hetkel kasutuses 8 GB kettaruumiga server, mis tähendab, et *mp3* failide salvestusruum on võrdlemisi väike. Kuigi laulud kustutatakse ära kui uus laul üles laetakse või kui ruum kinni läheb, siis on siiski võimalus, et kasutajad avavad piisavalt palju ruume ning laevad üles piisavalt laule, et kettaruum ära täita. Aga kuna selles lõputöös valmib pigem *proof-of-concept* ning ei ole prognoositud tohutut kasutajate voogu, siis hetkeline lahendus sobib rakenduse demonstratsiooniks. Joonisel (Joonis 4) on näidatud faili üleslaadimist ning andmete jagamist osapoolte vahel.



Joonis 4. Osapoolte töövood MP3 faili päringu korral

3.4.3 Heli sünkroniseerimine

Kuna projekti mõte oli kasutajatele mängida sünkroniseeritud helifaile, siis pidi võimaldama helifailide mängimist kindla algusajaga. Seetõttu salvestatakse serveri mällu aeg, millal fail mängima pandi, kui keegi vahetab laulu, olgu see siis kas YouTube lingiga või üleslaetud MP3 fail. Kasutaja liitumisel saadetakse talle laulu informatsioon koos algusajaga, et ta saaks õige päringu tekitada.

YouTube Iframe Player API'ga on kerge laul panna mängima kindlast punktist, peab ainult algusaja API kasutamisel kaasa andma. Salvestatud MP3 laulu mängimine kindlast ajapunktist on keerulisem. Andmete vaatepunktist on MP3 failid pikad baidijadad, mis töödeltakse heliks vastavalt selle helifaili bitkiirusele. Ehk on võimalik välja arvutada mitmendast baidist peab andmevoogu alustama, et muusika oleks teiste kasutajatega sünkroniseeritud. Helifaili bitikiirus kahjuks ei ole faili metaandmetes, seetõttu pidi kasutama *FFmpeg* teegi meetodit *ffprobe*, mis kogub ning muudab bitikiiruse ning muu informatsiooni kergesti kättesaadavaks. Ning lõpuks on kõik andmed olemas, et arvutada, mis baidist alates peab kasutajale andmevoona edastama helifaili.

3.5 Tagaliidese ühendamine eesliidesega

Selles peatükis kirjeldan samme, mida astuti, et ühendada tagaliides eesliidesega. Lõputöö raames koostatav eesliides ei pidanud olema eriti silmapaistev, põhieesmärk oli luua funktsionaalne eesliides, mis suutis tagaliidesele päringuid saata ning vastu võtta. Täiesti ilma disainita seda samuti ei saanud jätta, mistõttu tehti suhteliselt lihtne, kuid stiilitud eesliides.

3.5.1 Arendamine lokaalses keskkonnas

Toimiva eesliidese jaoks loodi kolm põhilist faili, HTML leht, JS skriptifail ning CSS stiilifail. Esialgu tekitati väga algelise kujundusega ning vähe stiilitud leht, millele lisati vormid HTML *form* märgendiga. Nendest väljadest hakati sisendit lugema, mis hiljem oli vaja saata ka serverile.

Seadistati üles ka lokaalses keskkonnas, pordil 3000 toimiv server, sest lehed oli vaja ühenduse korral serveri poolt saata kasutajale. Kui see tööle saadi, lisati ka mõned esialgsed päringud, nimelt sõnumi edastamine ning ruumidega ühinemine.

Arenduskäigus pandi failid mitu korda ka serverile jooksma, et kontrollida, kas seal ka töötab. Sellest protsessist tuligi välja, et Nginx'i seadistuses on vaja tõsta maksimaalset failisuurust, mis ühe *post* päringuga teha saab. Juhul kui serveri peal toimis midagi teistmoodi, siis asuti põhjust otsima ning kõik probleemid ka lahendati.

3.5.2 Ühendamine eesliidesega serveril

Hetkel saadab rakendus kliendile selle lõputöö käigus kirjutatud eesliidese. Serveril jooksev kood on kergesti muudetav, et oleks võimalik saata hoopis teised failid. Samuti prooviti eesliides teha võimalikult pakendatult, et oleks võimalik ümber tõsta asjalikuma eesliidese sisse. Kui kasutada selle projekti käigus valminu eesliidest, siis tuleb muuta ära üks rida koodi, mis teeb päringu serveri käest helifaili andmevoo saamiseks.

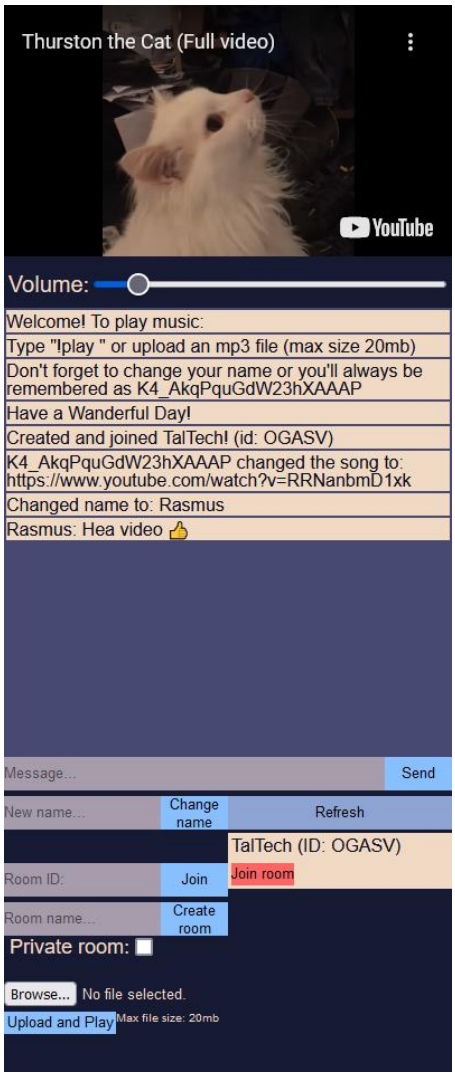
3.6 Muudatused vastavalt kasutajate tagasisidele

Kui eesliidese esimene versioon (Joonis 5) sai valmis, saadeti tuttavatele anonüümne küsitlus lehekülje eesliidese, selle diasini ning funktsionaalsuse kohta. Küsimusi esitati ruumisüsteemi (loomine, ühinemine ning avastamine) ja helimängija arusaadavuse kohta, elementide paiknemise kohta ning muude disainiosade, nagu värviskeem, kohta. Olenevalt küsimusest oli võimalik vastata kas „jah/ei", anda 1-5 skaala süsteemis punkte või kirjutada tekstiline vastus. Järgnevas lõigus toon välja ning analüüsin saadud tagasisidet.

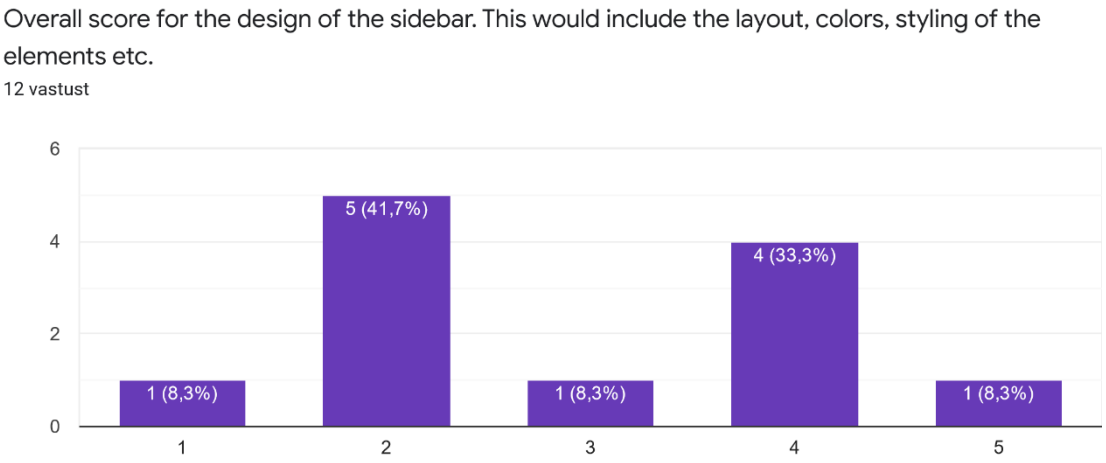
Esimene küsimus käsitles küljeriba laiust terve ekraani suhtes. Katsetatavas versioonis kattis see umbes 20% ekraanist. Kolm neljandiku vastajatest arvas, et see on hea suurus ning võiks jääda. Ruumide loomis- ning ühinemissüsteemi arusaadavus sai keskmiseks tulemuseks ~4.08, kuid üle poolte vastanutest (~54.5%) vastasid, et ei saanud avalike ruumide avastussüsteemist aru. Selge ülekaaluga oli tagasisides välja toodud, et võiks olla üks avalik ruum, mis on pidevalt aktiivne. Helimängimissüsteemi tagasisidest on võimalik näha, et MP3 failide üleslaadimise ning mängimise süsteemist on paremini aru saadud (keskmise tulemusega ~4.45) kui YouTube lingiga heli mängimise süsteemist (keskmise tulemusega ~4.33). Üks küsimus oli ka küljeribal esitatava informatsioonikoguse kohta, ehk küsiti, kas riba peal on liiga palju elemente. Küsimus

esitati skaalal 1-5 ning keskmiseks tulemuseks saadi ~3.18, kus 1 tähendab, et liiga palju ning 5, et ei ole liiga palju.

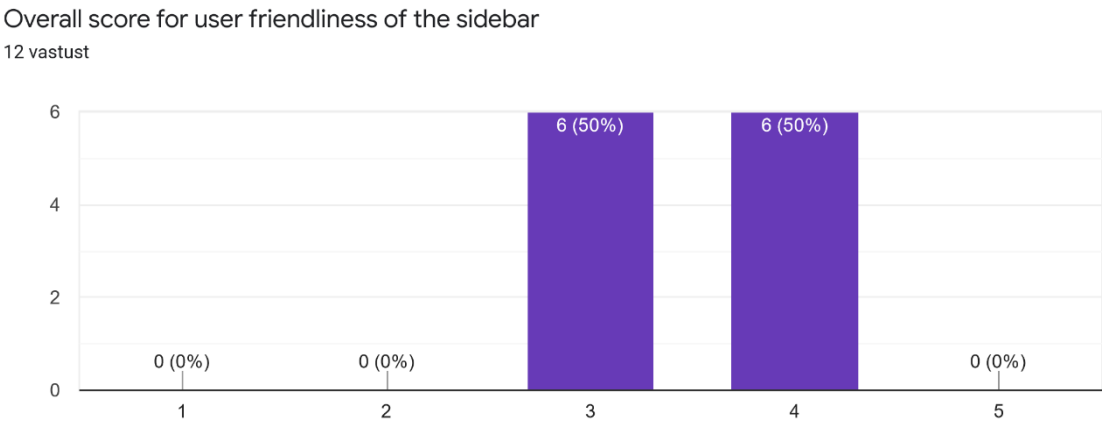
Seejärel esitati kaks üldisemat küsimust, mille tulemused on eraldi välja toodud ka joonistel. Kõigepealt küsiti arvulist hinnangut disaini kohta ning saadi tulemuseks keskmiselt ~2.92 (Joonis 6). See tähendas, et järgmine versioon eesliidesest kindlasti vajab uut disaini. Sõnalistest vastustest saadi teada, et värvid ja elementide paiknemine saaks palju parem olla. Seejärel oli küsimus kasutajasõbralikkuse kohta, mis sai keskmiseks tulemuseks 3.5 (Joonis 7). Ka siinkohal oli võimalik eesliidet paremaks teha ning anti ka selle jaoks konkreetseid soovitusi. Tagasisidet arvesse võttes kujundati uus eesliides, millele samuti tehti tagasiside voor. Täpsemalt sellest tulemuste peatükis.



Joonis 5. Eesliidese esimene versioon



Joonis 6. Tagasiside eesliidese disainile



Joonis 7. Tagasiside eesliidese kasutajasõbralikkusele

4 Tulemuste analüüs

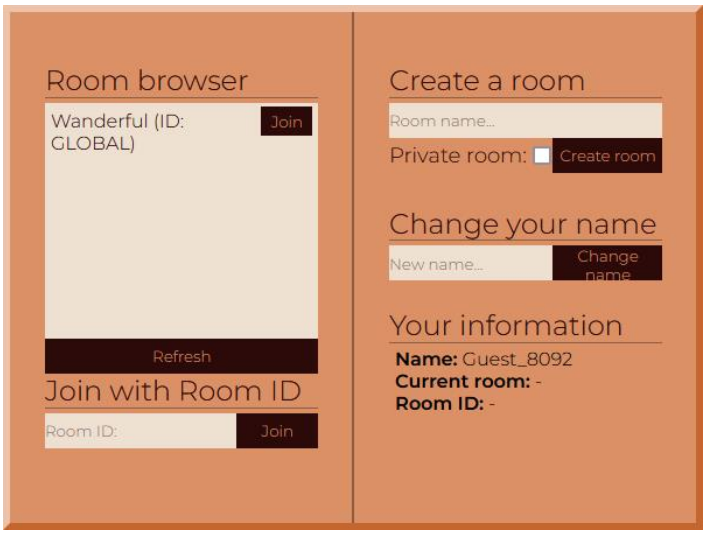
Lõputöö raames seadistati üles AWS'is majutatud veebiserver, millele suunatakse, kui minna domeenile *https://www.wanderful.day*[1]. Seejärel suunab Nginx tehtud päringu serveril jooksvale programmile, mis avab kasutajaga suhtluskanali sündmuste kuulamiseks. Sündmused, mida kuulatakse, on: MP3 faili üleslaadimise päring, MP3 faili allalaadimise päring, sõnumite saatmine, ruumiga ühinemine, hetkel mängiva laulu informatsiooni küsimine, nime vahetamine, ruumide nimekirja saamine, kindlas ruumis olevate kasutajate saamine ning ruumi loomine. Ühenduse tekkimisel saadetakse eesliidese failid, mis võimaldavad kasutajal päringuid teha ning vastu võtta.

4.1 Tagasiside rakenduse viimasele versioonile töö raames

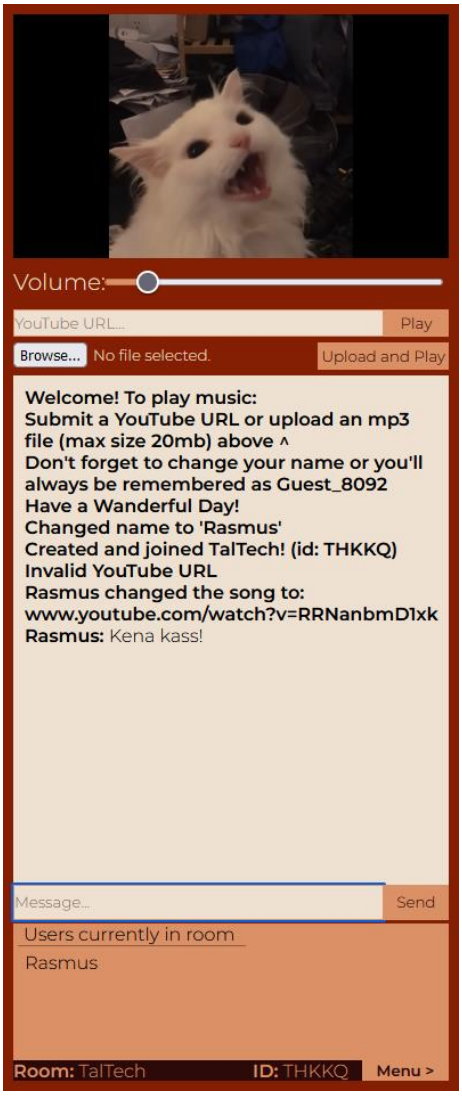
Vastavalt tagasisidele tehti eesliidesele peaaegu täielik ümberkujundus. Küljeriba eraldati kaheks erinevaks osaks (Joonised 8 ja 9). Värviskeem muudeti täielikult ära, kättesaadavaid andmeid lisati juurde, YouTube'st tulevad muusikat ei vahetata enam läbi tekstisisendi ning funktsionaalsus hajutati laiali kahe erineva elemendi sisse. Lisati ka üks avalik ruum, mis on pidevalt lahti ning kohe nähtav, kui lehele minna.

Uuendatud disainile tehti ka teine küsitluse voor, kus uuriti taaskord disaini ning kasutajasõbralikkuse kohta. Selles voorus said kõik proovijad aru kuidas avalike ruumide otsimine töötab. Helimängimissüsteemist saadi ka paremini aru. YouTube linkide mängimise keskmine punktisumma tõusis ~4.83 peale, mis on märgatav tõus eelnevast voorust. Arvamus uuest disainist tõusis ka ~4.33 peale, mis muidugi tähendab, et arenemisruumi veel on. Kasutajasõbralikkuse keskmine tulemus on pärast teist vooru 4.5. Selles tagasiside voorus olid veel mõningad kirjalikud soovitused, mida võetakse arvesse, kui projekti edasi arendatakse lõputöö väliselt.

[1] https://www.wanderful.day



Joonis 8. Menüüaken



Joonis 9. Täiendatud küljeriba

4.2 Arenguruum

Kuigi lõputöö raames valmis töötav põhifunktsionaalsusega rakendus, siis autor näeb, et seda saaks ka tohutult palju edasi arendada. Tagaliidese osas peaks uurima, kuidas rakendust suuremaks skaleerida juhul, kui on soov projektiga edasi tegeleda. Kui lähtuda esialgsest ideest, siis oleks rakendusele vaja lisada ka häälsuhtluskanalid, maksimaalsed ruumi suurused, kasutajatüübid (tavakasutaja, ruumi omanik), kasutajatüübile vastavad ruumisisesed õigused, sisselogimise võimalused ning kasutajaandmebaas ja tegelik andmebaas ruumiinformatsiooni hoiustamiseks. Tasub üle vaadata ka meedia esitusviisid, uurida Spotify ning SoundCloudi API'de lisamisvõimalusi ning ka uurida kas ja kuidas on võimalik otse kasutajatelt kasutajatele helifaile andevoogudena üle kanda, et vähendada serveri kettalt lugemise ja kirjutamise koormust.

5 Kokkuvõte

Eesmärgiks oli realiseerida veebirakendusele mõeldud tagaliides ning algeline eesliides. Lisaks sellele oli vaja üles seada veebiserver, domeenisuunamine, protsessihaldur ja muud veebiserveriga seonduvat. Töö käigus kaaluti ning prooviti erinevaid lähenemisi erinevatele probleemidele. Arendusfaasis leiti mitmeid vigu või takistusi, mille peale planeerimisfaasis ei osatud tullagi, mis seejärel said ka lahendatud.

Seatud eesmärgid ning tagaliidesele määratud nõuded said ka täidetud. Korraldati kaks tagasiside vooru, mille põhjal tehti projekis muudatusi. Esitatud projektis on veel palju arenguruumi ning kui autoril veel tulevikus huvi on antud teema vastu, siis võivad mõned ülaltoodud ideed ka realiseerimiseni jõuda.

Projekti käigus sai töö autor palju uusi teadmisi andmevoogude kohta, eriti helifailidega seotud voogude. Samuti oli uueks kogemuseks ehitada nullist põhimõtteliselt töötav veebirakendus.

Kasutatud kirjandus

[1] Microsoft Teams [Online] https://www.microsoft.com/et-ee/microsoft-teams/group-chat-software [Külastatud 08.05.2022]

[2] Turntable FM [Online] https://turntable.fm/ [Külastatud 08.05.2022]

[3] VRChat [Online] https://hello.vrchat.com/ [Külastatud 01.05.2022]

[4] Discord [Online] https://discord.com/ [Külastatud 01.05.2022]

[5] YouTube [Online] https://www.youtube.com/ [Külastatud 01.05.2022]

[6] Amazon Web Services [Online] https://aws.amazon.com/ [Külastatud 01.05.2022]

[7] NameCheap [Online] https://www.namecheap.com/ [Külastatud 01.05.2022]

[8] NodeJs [Online] https://nodejs.org/en/ [Külastatud 01.05.2022]

[9] Nginx [Online] https://www.nginx.com/ [Külastatud 01.05.2022]

[10] PM2 [Online] https://pm2.keymetrics.io/ [Külastatud 01.05.2022]

[11] Socket.IO [Online] https://socket.io/ [Külastatud 01.05.2022]

[12] Youtube Player API [Online] https://developers.google.com/youtube/iframe_api_reference [Külastatud 01.05.2022]

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

Mina, Rasmus Riismann

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose“ SÜNKRONISEERITUD MEEDIA ESITAMISE VEEBIRAKENDUS: WANDERFUL DAY“, mille juhendaja on Gert Kanter
   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.

01.05.2022

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