TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Marko Karpats 205905IADB

Telegram koosoleku abirakendus bot

Bakalaurusetöö

Juhendaja: Tiina Zingel

Tallinn 2023

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: Marko Karpats

24.04.2023

Annotatsioon

Käesolevas lõputöös tutvustatakse uue abirakenduse arendamist koosolekute korraldamiseks. Rakendus loodi eesmärgiga täita lüngad olemasolevates lahendustes ja pakkuda kasutajatele sujuvat ja turvalist kogemust. Rakendus on integreeritud Telegramiga ja omab sujuvat autoriseerimist, võimaldades kasutajatel luua ja hallata kohtumisi sõpradega ning saada soovituslikke tehinguid, et tasuda kõik võlad vastavalt tehtud tehingutele kohtumise ajal. Töös analüüsitakse põhjalikult probleemi ja teiste lahenduste nõrkusi, samuti antakse ülevaade arendusprotsessist, sealhulgas arhitektuurist, ning nii veebiteenuse kui ka kliendirakenduse lahendustest.

Kuigi arendusprotsessi käigus ei ole kõik funktsioonid valmis saanud, pakub rakendus enamikku analüüsis täpsustatud nõuetest ja on kättesaadav avaliku Telegram-boti kaudu. Lõputöö paneb tugeva aluse rakenduse edasiseks arendamiseks, võimaldades testimist, struktureerimist ja puhta koodi põhimõtete järgimist.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 30 leheküljel, 6 peatükki, 16 joonist.

Abstract

Telegram Meeting Helper Bot

This thesis presents the development of a new helper application for organizing meetings. The application was created with the goal of filling gaps in existing solutions and providing a seamless, secure experience for users. The application is integrated with Telegram and features seamless authorization, allowing users to create and manage meetings with friends and receive suggested transactions to settle all debts according to made transactions during the meeting. The work provides a thorough analysis of the problem and weaknesses of other solutions, as well as an overview of the development process, including the architecture, and solutions of both the web service and client application.

While not all functionalities were completed during the development process, the application provides most of the requirements specified in the analysis and is accessible through a public Telegram bot. The thesis lays a strong foundation for future development of the application by encouraging testing, structuring, and adhering to clean code principles.

The thesis is in estonian and contains 30 pages of text, 6 chapters, 16 figures.

Sisukord

Jooniste loetelu

Mõistete ja lühendite selgitus

API	Application Programming Interface
Backend	Serverrakendus
CSS	Cascading Style Sheets
DOM	Document Object Model
ERD	Entity Relationship Diagram
Frontend	Klientrakendus
HTTP	Hyper Text Transfer Protocol
I/O model	An Input-Output model
JSON	JavaScript Object Notation
JVM	Java Virtual Machine
RSVP	"Répondez s'il vous plaît", mis tähendab prantsuse keeles "palun vastake"
SQL	Structured Query Language
UI	User Interface
URL	Uniform Resource Locator

1 Sissejuhatus

Kiire elutempo on muutnud sõprade isikliku kokkusaamise keeruliseks. Kuna vahemaad suurenevad ja ajakava on keerulisem kooskõlastada, on sõpradega kohtumine muutunud keeruliseks ülesandeks. Sotsiaalmeedia ja sõnumirakendused, nagu Telegram, on aga lihtsustanud ühenduse hoidmist hoolimata geograafilistest takistustest. Kuna Telegrami kasutamine sotsiaalseks suhtluseks on suurenenud, on vaja platvormi, mis võimaldab sõpradel hõlpsasti planeerida ja korraldada omavahelisi üritusi.

Vastuseks sellele vajadusele on käesoleva lõputöö eesmärk töötada välja veebirakendus, mis aitab korraldada sõprade kohtumisi Telegramis. Rakendus pakub platvormi, kus sõbrad saavad luua ja hallata sündmusi, arutada sündmuse üksikasju ja kutsuda teisi sõpru osalema. Veebirakendusel on ka funktsioon, mis võimaldab sõpradel hääletada sündmuse üksikasjade, näiteks asukoha, kuupäeva ja kellaaja üle, et optimeerida sündmuse kava kõigi osalejate eelistuste järgi.

Käesoleva lõputöö hindamine põhineb WebAppi rakenduse edukal arendamisel ja selle tõhususel Telegrami sõprade ürituste korraldamisel. Selle lõputöö tulemused on kasulikud üksikisikutele ja organisatsioonidele, kes soovivad parandada oma sotsiaalse koordineerimise ja sündmuste planeerimise kogemusi.

2 Ülevaade probleemist

Koosolekute või ürituste korraldamine on alati olnud keeruline ja nõudnud palju aega ja pingutust. Kuigi selle protsessi lihtsustamiseks on saadaval palju rakendusi, on neil kõigil omad piirangud. Alates kalendrirakendustest kuni spetsiaalsete ürituste planeerimise tööriistadeni ei tundu ükski neist pakkuvat täielikku lahendust.

Üks suurimaid probleeme nende rakenduste kasutamisel on vajadus vahetada pidevalt platvorme, et hallata sündmuse planeerimise protsessi erinevaid aspekte. See võib olla aeganõudev ja segadust tekitav, eriti kui püütakse kooskõlastada mitme inimesega.

Teine probleem on paljude nende rakenduste vähene kohandatavus ja paindlikkus. Need on sageli loodud konkreetset töövoogu või kasutusviisi silmas pidades ning neid ei pruugi olla lihtne kohandada konkreetse ürituse ainulaadsete vajaduste rahuldamiseks.

Lisaks nõuavad paljud neist rakendustest kasutajatelt kontode loomist või isikliku teabe jagamist, mis võib mõnede kasutajate jaoks olla eemaletõukav. Samuti tasub märkida, et paljud neist rakendustest ei ole tasuta, mis võib mõne kasutaja jaoks olla takistuseks nende kasutuselevõtmisel.

2.1 Metoodika

Üks olulisemaid samme eduka rakenduse loomisel on lisaks alguses tehtavale hoolikale analüüsile ka testimine. Oluline on veenduda, et funktsionaalsus töötab nõutaval viisil ja et kasutaja saab intuitiivselt aru, kuidas rakendus töötab. Iga väike detail, mida kasutaja rakenduse esmakordsel kasutamisel märkab, mõjutab oluliselt seda, kuidas ta toote suhtes suhtub.

Selleks antakse rakendusele juurdepääs väikesele ringile inimesi. Nende tagasiside põhjal on võimalik leida valmis programmis kõige ilmsemad puudused ja parandada need enne, kui sihtrühm hakkab programmi kasutama.

2.2 Eksisteerivad lahendused

Selles jaotises on kirjeldatud turul olemasolevaid lahendusi, mis käsitlevad sama probleemi. On analüüsitud nende tugevusi ja nõrkusi, et teha kindlaks täiustamisvõimalused, mida saab teha arendataval rakendusel.

2.2.1 Doodle

Doodle on ajaplaneerimisplatvorm, mis pakub lihtsat ja intuitiivset kasutajaliidest, mis teeb küsitluste loomise ja neile vastamise kasutajatele lihtsaks. Selle kohandatavad valikud, sealhulgas kuupäev ja kellaaeg, võimaldavad kasutajatel kohandada küsitlusi vastavalt oma konkreetsetele vajadustele. Doodle integreerub ka populaarsete kalendriplatvormidega, nagu Google Calendar ja Microsoft Outlook, muutes ürituste planeerimise lihtsamaks. Automaatsed meeldetuletused osalejatele vähendavad veelgi koosolekute või ürituste vahelejäämise tõenäosust. Doodle on kättesaadav mitmel platvormil, sealhulgas veebirakendus, mobiilirakendus ja brauseripikendus.

Siiski ei pruugi Doodle'i piiratud funktsioonid olla piisavad keeruliste sündmuste jaoks ning see ei paku reaalajas uuendusi ega palju võimalusi küsitluse üldise välimuse ja tunnetuse kohandamiseks. Lisaks ei jälgi see, kes on küsitlusele vastanud ja kes mitte, mis võib olla osalemise jälgimisel keeruline. Üldiselt on Doodle kasulik vahend mitme osalejaga koosolekute või ürituste korraldamiseks, kuid selle piirangutega tuleks arvestada keerukamate ürituste planeerimisel. [1]

2.2.2 Google Calendar

Google Calendar on paindlik ja võimas ajaplaneerimisvahend, mis integreerub sujuvalt teiste Google'i toodetega, muutes sündmuste ja koosolekute haldamise lihtsaks. Selle kohandatavad teavitused ja koostööfunktsioonid, näiteks võimalus vaadata teiste inimeste kalendreid ja kutsuda teisi üritustele, teevad sellest kasuliku vahendi mitme osalejaga koosolekute või ürituste korraldamiseks. Lisaks on mobiilirakendus kasutajasõbralik ja võimaldab üritusi ja kohtumisi hõlpsasti hallata ka liikvel olles. Google kalendrit saab kasutada ka muudel eesmärkidel kui koosolekute planeerimine, näiteks tähtaegade jälgimiseks või isiklike sündmuste planeerimiseks.

Siiski on Google kalendril mõningaid piiranguid kohandamisvõimaluste osas, eriti mis puudutab kalendri üldist välimust ja tunnetust. Lisaks ei paku see sisseehitatud küsitlus- või hääletamisfunktsiooni, mis võib olla piiranguks, kui üritatakse planeerida mitme osalejaga

sündmusi. Samuti võivad tekkida ajavööndiprobleemid, kui korraldate üritusi, mille osalejad asuvad erinevates ajavööndites. Kuigi Google Calendar on kasutajasõbralik, võib platvormiga alles hiljuti tutvunute jaoks siiski tekkida õppimisraskus, eriti kui tegemist on edasijõudnute funktsioonidega.

Üldiselt on Google Calendar võimas ja paindlik vahend, mis võib olla kasulik koosolekute või ürituste korraldamisel, eriti kui seda kasutatakse koos teiste Google'i toodetega. Selle tugevate külgede hulka kuuluvad sujuv integratsioon teiste Google'i toodetega, kohandatavad teavitused ja koostööfunktsioonid. Siiski tuleks selle piirangutega arvestada, kui planeeritakse keerulisemaid üritusi, näiteks selliseid, mis nõuavad küsitlust või hääletamist konkreetsetel kuupäevadel või kellaaegadel. [2]

2.2.3 Meetup

Meetupi tugevate külgede hulka kuuluvad keskendumine kogukonna loomisele, suur kasutajaskond, täiustatud otsingu- ja filtreerimisvõimalused ning jõulised sündmuste loomise ja haldamise vahendid. Kasutajad saavad otsida üritusi ja gruppe mitmesuguste kriteeriumide alusel ning luua üritusi koos RSVP jälgimise ja ürituste kirjelduse võimalustega. Lisaks on Meetupi mobiilirakendus kasutajasõbralik ja võimaldab üritusi hõlpsasti hallata ka liikvel olles.

Meetupi mõned nõrgad küljed on siiski piiratud kohandamisvõimalused, kallid lisafunktsioonid, piiratud kontroll osalejate üle, piiratud toetus korduvatele üritustele ja potentsiaalne õppimiskõver uutele kasutajatele. Kasutajatel ei pruugi olla nii palju kontrolli ürituse osalejate üle ja täiustatud funktsioonid nõuavad tasulist tellimust, mis ei pruugi olla kõigi kasutajate jaoks teostatav. Lisaks, kuigi Meetup on kasutajasõbralik, võib uute kasutajate jaoks olla keeruline õppida kasutama keerulisemaid funktsioone. [3]

2.2.4 Facebook Events

Facebooki üritused on populaarne rakendus koosolekute ja ürituste korraldamiseks. Facebook Events'i tugevate külgede hulka kuuluvad selle tohutu kasutajaskond, integratsioon Facebookiga, lihtne sündmuste loomine ja RSVP jälgimine ning mitmesugused reklaamivõimalused. Miljonite kasutajatega Facebook Events teeb ürituste loomise ja reklaamimise laiale publikule lihtsaks ning selle integratsioon Facebookiga võimaldab lihtsat ürituste jagamist ja reklaamimist isiklikes

profiilides. Kasutajad saavad luua ja hallata sündmusi, mille üksikasjad, nagu sündmuse kirjeldus, kuupäevad, kellaajad ja asukohad, on kohandatavad. Vastuste jälgimise abil on lihtne näha, kes on ürituse kutsele vastanud, ja hallata osalejate nimekirju vastavalt sellele ning Facebook Events pakub erinevaid reklaamivõimalusi, sealhulgas tasulist reklaami ja orgaanilist jagamist, et suurendada ürituse nähtavust.

Siiski on Facebook Eventsil ka mõned nõrgad küljed. Üks peamisi probleeme Facebooki ürituste kasutamisel on privaatsus, nagu kõigi Facebooki funktsioonide puhul. Mõnedel kasutajatel võivad Facebooki ürituste kasutamisel tekkida mured andmete kogumise ja privaatsuse pärast. Lisaks, kuigi Facebook Events pakub põhilisi kohandamisvõimalusi, nagu näiteks kaanepildi lisamine, on kasutajatel piiratud kontroll oma sündmuse lehekülje üldise kujunduse ja brändi üle. Facebook Events pakub ka piiratud kontrolli osalejate üle ning kasutajad ei pruugi olla võimelised piirama osalejate arvu või jälgima osavõtjate arvu suuremate külaliste nimekirjadega ürituste puhul. Facebooki ürituste mobiilirakendus on võrreldes töölauaversiooniga piiratud funktsionaalsusega, mis võib muuta ürituste haldamise liikvel olles keeruliseks. Kuna Facebookis toimub igal ajal nii palju üritusi, võib olla keeruline üritust esile tõsta ja osalejaid kohale meelitada. [4]

2.2.5 Trello

Trello on populaarne projektijuhtimise rakendus, mida saab kasutada koosolekute ja ürituste korraldamiseks. Trello tugevate külgede hulka kuuluvad kasutajasõbralik kasutajaliides, paindlikkus, koostööfunktsioonid ja kohandamisvõimalused. Tänu lohistamisliidesele võimaldab Trello hõlpsasti luua ja hallata ülesandeid ja projekte, sealhulgas korraldada ja hallata koosolekuid ja üritusi. Kasutajad saavad iga projekti või sündmuse jaoks luua tahvleid ning luua nimekirju ja kaarte, et jälgida üksikasju, nagu päevakorrad, ülesannete nimekirjad ja tähtajad. Trello pakub ka paindlikke kohandamisvõimalusi, sealhulgas võimalust lisada kaartidele manuseid, kontrollnimekirju ja kommentaare ning määrata ülesandeid konkreetsetele meeskonnaliikmetele. Trello koostööfunktsioonid võimaldavad meeskonnaliikmetel suhelda ja teha projektide osas koostööd reaalajas, mistõttu on see kasulik vahend ürituste planeerimiseks.

Siiski on Trellol ka mõned nõrgad küljed. Üks Trello peamisi piiranguid on see, et see on mõeldud eelkõige ülesannete haldamiseks, mitte konkreetselt sündmuste planeerimiseks. Kuigi seda saab kasutada ürituste ja koosolekute korraldamiseks, ei pruugi sellel olla kõiki funktsioone, mida

pakuvad teised spetsiaalselt ürituste planeerimiseks mõeldud rakendused, näiteks RSVP jälgimine või integratsioon kalendritega. Samuti võib Trello muutuda üle jõu käivaks, kui hallata on liiga palju kaarte, nimekirju või tahvleid, mis võib muuta suurte või keeruliste ürituste korraldamise keeruliseks. Lisaks, kuigi Trello pakub palju kohandamisvõimalusi, võib iga tahvli, kaardi või nimekirja seadistamine ja kohandamine olla aeganõudev, mis ei pruugi olla praktiline kasutajatele, kes peavad üritusi kiiresti või tõhusalt korraldama. Lõpuks võib Trello kasutajaliides olla visuaalselt vähem ahvatlev kui teised projektijuhtimise vahendid, mis ei pruugi mõnede kasutajate jaoks olla nii kaasahaarav või inspireeriv. [5] [6]

2.3 Pakutava lahenduse eelised

Esiteks on märkimisväärne eelis see, et rakendust saab kasutada populaarse Telegrami sõnumitooja sees. See tähendab, et kasutajad saavad hõlpsasti korraldada üritusi vestluses, mida nad juba aktiivselt kasutavad, selle asemel, et liikuda eri platvormide vahel.

Lisaks, kuna Telegramil on juba suur kliendibaas, jõuab rakendus hõlpsasti paljude potentsiaalsete kasutajateni. Lisaks on eeliseks ka asjaolu, et lahendus ei nõua täiendavat registreerimist, kuna see lihtsustab kasutajakogemust ja vähendab uute kasutajate jaoks sisenemisbarjääri.

Oluliseks eeliseks teiste lahendustega võrreldes võib olla ka funktsioon, mis näitab, kui palju iga külaline pärast ürituse toimumist võlgu on. See teeb osalejatele lihtsaks teada saada, kui palju nad peavad üksteisele tagasi maksma, ning võib vältida arusaamatusi ja konflikte.

Võimalus jälgida sündmuse nõuete edenemist on veel üks kasulik funktsioon, sest see aitab kasutajatel jääda organiseerituks ja olla kursis kõigega, mida tuleb teha. Võimalus määrata igale nõudele vastutav isik võib samuti olla kasulik, sest see võib tagada, et ülesanded täidetakse õigeaegselt ja tõhusalt.

3 Loodava rakenduse analüüs

Selles osas uurime rakenduse funktsionaalseid ja mittefunktsionaalseid nõudeid, tuues välja konkreetsed funktsioonid ja võimalused, mida see peaks pakkuma. Lisaks arutame arendusprotsessi jaoks valitud tehnoloogiaid, sealhulgas programmeerimiskeeli, raamistikke ja vahendeid, mida kasutati nende nõuete tõhusaks ja tulemuslikuks täitmiseks.

3.1 Funktsionaalsed nõuded

Funktsionaalsed nõuded on konkreetsed funktsioonid ja võimalused, mis peavad rakendusel olema, et see täidaks oma eesmärki. Siin on funktsionaalsed nõuded, mida sündmuse korraldamise rakendus vajab:

- Rakendus peaks võimaldama kasutajatel sujuvalt registreeruda/sisselogida, kasutades andmeid, mida Telegram pakub veebirakenduse jaoks.
- Kasutajatel peaks olema võimalik luua ja seadistada koosolekut.
- Kasutajatel peaks olema võimalik luua ja seadistada koosoleku sündmusi, sealhulgas määrata kuupäev, kellaaeg ja asukoht.
- Rakendus peaks võimaldama kasutajatel luua ja hallata iga sündmusega seotud ülesandeid/nõudeid, sealhulgas määrata ülesandeid konkreetsetele kasutajatele ja jälgida seda täitmist.
- Kasutajal peaks olema võimalus saata teistele Telegrammi kasutajatele kutseid Telegrammi kaudu.
- Koosoleku organiseerijal peaks olema võimalik hallata külaliste nimekirja, sealhulgas eemaldada, anda või tagasi võtta õiguseid.
- Kui üritusel ajal tuleb teha makset, peaks rakendus võimaldama kasutajatel panna kirja tehtud makseid ja pidada arvestust makseajaloo üle.

- Kasutajal peab olema võimalus pärast üritust näha kellele ta võlgneb mingi ürituse nõude eest.

3.2 Mittefunktsionaalsed nõuded

Mittefunktsionaalsed nõuded viitavad omadustele, mis peaksid rakendusel olema, et tagada selle tõhus ja tulemuslik töö. Siin on mõned mittefunktsionaalsed nõuded, mida tuleks arvesse võtta.

Rakendus peaks olema turvaline, et vältida volitamata juurdepääsu andmetele või muule tundlikule teabele. Tuleks rakendada turvameetmeid, et kaitsta kasutajate andmeid, eriti nende isikuandmeid, volitamata juurdepääsu eest.

See peaks olema usaldusväärne ja alati kättesaadav, et kasutajad saaksid sellele alati juurde pääseda, kui nad seda vajavad. See tähendab, et rakendus peaks olema projekteeritud nii, et seisakud ja katkestused oleksid minimaalsed. Mis tähendab, et arhitektuur peaks olema kavandatud nii, et vigu oleks lihtne parandada ja uusi funktsioone lisada.

Samuti peaks see olema võimeline töötlema suurt arvu kasutajaid ja sündmusi. Rakendus peaks olema skaleeritav, et tulla toime kasvava kasutajaskonna ja suurenenud kasutusega, ilma et see mõjutaks jõudlust.

Üks olulisemaid on see, et see peaks olema kasutajasõbralik ja kergesti kasutatav igat tüüpi kasutajatele. See tähendab, et rakendusel peaks olema intuitiivne kasutajaliides ja selles peaks olema lihtne navigeerida. See hõlmab ka seda, et rakendus peaks olema ühilduv ja toimima erinevates seadmetes umbes samamoodi.

Kasutajasõbralikumaks muutmiseks peaks rakendus olema hästi toimiv, kiire ja tundlik. See peaks olema võimeline töötlema suuri andmemahte ilma aeglustumise või kokkupõrketa.

3.3 Tehnoloogia valik

3.3.1 Andmebaas

Usaldusväärse andmebaasi haldussüsteemi valimisel on mitmeid populaarseid võimalusi, sealhulgas PostgreSQL, MySQL, Oracle ja Microsoft SQL. Igal neist andmebaasidest on oma ainulaadsed omadused ja eelised, mistõttu on oluline hoolikalt kaaluda, milline variant sobib teie vajadustele kõige paremini. Pärast põhjalikku hindamist selgub, et PostgreSQL paistab konkurentide seast silma.

Kuigi MySQL ja Microsoft SQL on mõlemad laialdaselt kasutusel ja hinnatud, on neil mõlemal mõningaid piiranguid seoses skaleeritavuse ja keeruliste andmetüüpidega. Oracle seevastu on võimas valik, kuid selle rakendamine ja hooldus võib olla kallis, mistõttu on see väiksematele või alustavatele projektidele vähem kättesaadav.

PostgreSQL pakub aga mitmeid eeliseid, mis muudavad selle paljude arendajate ja ettevõtete jaoks selgeks valikuks. Eelkõige on PostgreSQL uskumatult mitmekülgne ja suudab käsitleda väga erinevaid andmetüüpe ja keerulisi päringuid. See pakub ka suurepärast skaleeritavust ja jõudlust, mistõttu on see suurepärane valik suure liiklusega veebisaitide ja rakenduste jaoks. Lisaks on PostgreSQL avatud lähtekoodiga, mis tähendab, et seda saab tasuta kasutada ja seda saab kohandada vastavalt teie konkreetsetele vajadustele.

Üks peamisi omadusi, mis eristab PostgreSQLi konkurentidest, on selle tugev tugi täiustatud SQL-funktsioonidele. See lihtsustab keerukate päringute kirjutamist ja andmeanalüüsi ülesannete täitmist, mis on eriti oluline ettevõtetele, kes tuginevad otsuste tegemisel suuresti andmete sisestusele.

Teine PostgreSQLi eelis on selle tugev arendajate ja kasutajate kogukond, mis tähendab, et probleemide lahendamiseks, optimeerimiseks ja kohandamiseks on saadaval hulgaliselt ressursse. See võib säästa arendajate aega ja vaeva, võimaldades neil keskenduda suurepäraste rakenduste loomisele, mitte andmebaasiga seotud probleemidega tegelemisele.

Kokkuvõttes on PostgreSQL usaldusväärne, paindlik ja võimas valik ettevõtetele ja arendajatele, kes otsivad andmebaasi haldussüsteemi, mis saab hakkama keeruliste andmetüüpidega ja toimib hästi suure liikluse korral. Selle avatud lähtekoodiga olemus, ulatuslik tugi täiustatud SQL-funktsioonidele ja tugev kogukond teevad sellest suurepärase valiku kõigile, kes soovivad luua kvaliteetseid rakendusi ja teenuseid. [7] [8]

3.3.2 Serverrakendus

Selles osas võrdleme erinevaid serverrakenduse-arendusraamistikke, sealhulgas ASP.NET-i koos Entity Framework-iga, Node.js-i, Java Springi ja Pythoni Django raamistikke. Backend raamistik on tööriistade ja raamatukogude kogum, mis aitab arendajatel luua veebirakendusi ja -teenuseid. See annab struktuuri koodi korraldamiseks, päringute ja vastuste käsitlemiseks ning andmebaasidega suhtlemiseks. Igal raamistikul on oma ainulaadsed omadused ja eelised ning selle valik, millist raamistikku kasutada, sõltub projekti konkreetsetest vajadustest. Nende raamistike võrdlemise ja vastandamise abil püüame anda ülevaate nende tugevatest ja nõrkadest külgedest ning aidata arendajatel teha teadlikke otsuseid selle kohta, millist raamistikku kasutada.

Keerulise backend'i arendamine nõuab palju kohandamist, optimeerimist ja turvameetmeid. Kuigi Django ja Node.js saavad hakkama keerulise backend-arendusega, on neil omad piirangud. Django kasutatakse peamiselt sisuhaldussüsteemide, sotsiaalmeediaplatvormide ja e-kaubanduse veebisaitide ehitamiseks. Selle eeltäidetud komponendid, näiteks kasutaja autentimine ja haldusliides, muudavad rakenduste kiire ehitamise lihtsaks. Kui aga on vaja rakendada kohandatud äriloogikat või andmetöötlust, võivad arendajad selle tihedalt seotud arhitektuuri tõttu raskustesse sattuda. Sarnaselt kasutatakse Node.js-i laialdaselt reaalajas rakenduste, näiteks vestlusrakenduste või online-mänguplatvormide ehitamiseks. Selle mitteblokeeriv I/O-mudel ja sündmustepõhine arhitektuur teevad sellest suurepärase valiku skaleeritavate rakenduste loomiseks. Siiski võib arendajatel olla raske säilitada koodi kvaliteeti ja jõudlust selle asünkroonse olemuse tõttu.

Django või Node.js-i kasutamisel on veel üks probleem, et nii Python kui ka JavaScript on dünaamiliselt tüpiseeritud keeled, mis võib põhjustada ootamatuid erandeid töö ajal. See tähendab, et vigu ei pruugi tabada enne tööaega, põhjustades rakenduse seisakuid või ootamatut käitumist. Seevastu staatiliselt tüpiseeritud keeled, nagu C# ja Java, püüavad vead kinni juba kompileerimise ajal, mis muudab koodi kvaliteedi säilitamise lihtsamaks ja vähendab tööaegsete erandite riski.

Seetõttu on keerulise backend'i arendamisel C# koos ASP.NET-i ja Entity Framework-iga ning Java koos Spring Framework-iga kaks kõige sobivamat võimalust. Nende raamistike kasutamise üks peamisi eeliseid on nende tugev tüübisüsteem, mis lihtsustab vigade püüdmist ja vähendab ootamatute erandite arvu töö ajal. Lisaks pakuvad mõlemad raamistikud laia valikut raamatukogusid ja tööriistu, mis toetavad arendajaid keerulise äriloogikaga suuremahuliste rakenduste loomisel.

ASP.NET on küps ja väljakujunenud raamistik, mis pakub võimsaid vahendeid ettevõtlusklassi veebirakenduste arendamiseks. See on ehitatud .NET raamistiku peale, mis pakub töökindlat ja turvalist platvormi veebirakenduste loomiseks. Entity Framework-iga saavad arendajad hõlpsasti kaardistada andmebaasi tabelid C# objektidele, mis lihtsustab tööd andmebaasidega ja andmete haldamist.

Teisalt on Spring Framework kerge ja paindlik raamistik, mis pakub laias valikus mooduleid, mis aitavad arendajatel luua keerulisi rakendusi. See on ehitatud Java virtuaalmasina (JVM) peale, mis teeb rakenduste kasutuselevõtu ja käivitamise lihtsaks paljudel platvormidel. Spring pakub ka mitmeid vahendeid ja raamatukogusid andmebaasidega töötamiseks, samuti tuge erinevatele sõnumivahetussüsteemidele, vahemälu salvestamisele ja muule.

Siiski tasub märkida, et nii C# kui ka Java on kompileeritud keeled, mis tähendab, et arendus ja silumine võib võtta kauem aega kui dünaamiliselt tüpiseeritud keelte, näiteks Pythoni või JavaScripti puhul. Lisaks võib nende raamistike õppimine olla järsem arendajatele, kes on vähem tuttavad objektorienteeritud programmeerimise põhimõtetega. [9] [10]

3.3.3 Klientrakendus

Rakenduse loomise protsessis on kliendipoolel oluline roll sujuvas kasutajakogemuses. JavaScript on pikka aega olnud domineeriv programmeerimiskeel frontend-arenduses, pakkudes laia valikut funktsioone ja funktsioone. Kuid tõhusa ja tulemusliku frontend'i loomisel on sama oluline valida õiged arendusvahendid. Kuigi frontend-arenduseks võib kasutada ka tavalist JavaScripti, võib see pikendada arendustsükleid ja vähendada tootlikkust. Nende piirangute ületamiseks kasutab enamik kaasaegseid kliendirakendus arendamise kiirendamiseks selliseid raamistikke nagu Angular, Vue, React ja teisi. Need raamistikud pakuvad kindlaid vahendeid ja teeke, mis lihtsustavad

arendusprotsessi ja võimaldavad arendajatel hõlpsasti ehitada keerulisi rakendusi. Seetõttu võib sobiva raamistiku valimine mõjutada oluliselt rakenduse kvaliteeti ja jõudlust.

Angular on populaarne valik frontend'i arendamiseks. Angulari üks suurimaid eeliseid on see, et tegemist on täieõigusliku raamistikuga. See tähendab, et see on varustatud mitmesuguste sisseehitatud funktsioonidega, näiteks võimsa mallide süsteemi, kahesuunalise andmesidumise ja sõltuvussüstimisega. Need funktsioonid muudavad keeruliste, suure hulga komponentidega rakenduste loomise lihtsaks ja vähendavad arendajate poolt kirjutatava koodi hulka.

Angulari teine tugevus on selle tugev tüpiseeritud süntaks. TypeScripti abil, mida Angulari projektides vaikimisi kasutatakse, saavad arendajad varakult vigu tabada ja tagada, et nende kood on korrektne juba enne selle käivitamist. See võib arengu ajal märkimisväärselt aega kokku hoida ja aitab vältida vigade jõudmist tootmisse. Kuna aga TypeScript teisendatakse käivitamisajal JavaScriptiks, võib see siiski tekitada palju ootamatuid erandeid.

Kuid see kasu ei kaalu üles Angulari üht nõrkust. Ja see on selle jõudlus. See juhtub suure hulga DOM-manipulatsioonide tõttu, mis on Angulari kasutamisel vajalikud. Rakendus võib kiiresti muutuda aeglaseks ja reageerimatuks, kui seda ei ole õigesti optimeeritud.

Ka Vue võib olla teine elujõuline võimalus. Üks Vue tugevusi on selle kasutamise lihtsus. Sellel on suhteliselt väike õppimiskõver, mistõttu on lihtne kiiresti alustada UI-komponentide ehitamist. Lisaks pakub Vue virtuaalse DOMi rakendamine head jõudlust ja tõhusaid uuendusi, kuna virtuaalse DOMi muudatusi rakendatakse reaalsele DOMile ainult siis, kui see on vajalik.

Siiski on Vue'l ka mõned nõrgad küljed, millega tuleb arvestada. Üks selline nõrkus on selle piiratud ökosüsteem, mille kogukond on väiksem ja ressursse on vähem võrreldes teiste populaarsete raamistikega nagu React. See võib muuta probleemide lahenduste leidmise või toetuse saamise keerulisemaks. Teine nõrkus on Vue'i piirav mallide süsteem, mis võib muuta keerulisemate UI-komponentide loomise keeruliseks. Kuigi Vue'i mallide süsteem on algaja jaoks kergemini ligipääsetav, võib see muutuda piiravaks, kui püütakse luua keerulisemaid funktsioone.

Reactil on teiste raamistike ees mitmeid selgeid eeliseid, mistõttu on see arendajate seas populaarne valik. Üks peamine eelis on Reacti virtuaalne DOM, mis võimaldab suurt jõudlust, vähendades tegelike muudatuste arvu reaalses DOMis. Selle tulemuseks on palju kiirem

renderdamisprotsess ja parem üldine jõudlus. Lisaks on Reactil suur ja aktiivne arendajate kogukond, kes panustavad paljudesse teeke, mis teeb sellest mitmekülgse ja paindliku raamistiku igasuguste rakenduste jaoks.

Teine Reacti eelis on selle ühilduvus TypeScriptiga, mis aitab vältida ootamatuid tööaja vigu ja parandada koodi üldist kvaliteeti. Lisaks võimaldab Reacti JSX-süntaks tänu HTML-süntaksi sarnasusele hõlpsasti luua keerulisi kasutajaliidese komponente. See omadus võib aidata vähendada ka trükivigade ja muude vigade tõenäosust koodis, enne kui see jõuab isegi käivitamisele.

Reacti üks potentsiaalne puudus on selle sõltuvus funktsionaalsetest programmeerimisparadigmadest, mis võib olla väljakutse arendajatele, kes on harjunud traditsioonilisemate objektorienteeritud mudelitega. Kuid tänu suurele kogenud arendajate kogukonnale ei ole üldjuhul raske leida abi ja tuge nende esialgsete takistuste ületamiseks. Samuti on tänu suurele kogukonnale saadaval palju raamatukogusid Reacti jaoks. Kokkuvõttes teevad Reacti arvukad tugevused ja kasvav populaarsus sellest suurepärase valiku frontend-arenduseks. [11] [12] [13]

3.3.4 Kliendirakenduse lisa raamistikud

Veebilehe elementide stiili arendamiseks kasutatakse selles rakenduses Tailwind CSS-i. Tailwind CSS on kasuliku CSS-raamistik, mis pakub komplekti eeldefineeritud klasse, et luua kiiresti järjepidev ja tundlik kasutajaliides. Tailwind CSSi kasutamisel Reacti veebirakenduses võib täheldada mitmeid eeliseid.

Esiteks parandab see arenduse kiirust, kuna Tailwind CSS pakub suurt hulka eeldefineeritud klasse, mida saab kasutada kasutajaliidese kiireks loomiseks.

Samuti vähendab see CSS-i paisumist, sest Tailwind genereerib CSS-i ainult nende klasside jaoks, mida veebirakenduses tegelikult kasutatakse, mille tulemuseks on väiksemad CSS-failid ja kiirem lehe laadimisaeg.

Kuid Tailwind CSS'i ebaõige kasutamine võib põhjustada dubleeritud stiilide paljusust, mis põhjustab raskusi koodi lugemisel ja parandamisel. Seetõttu on selle tööriista kasutamisel väga oluline olla ettevaatlik, et optimeerida selle võimalikku kasu. [14]

Framer Motion on populaarne teek komponentide animeerimiseks Reacti veebirakendustes. See pakub mitmeid eeliseid, mis võivad parandada veebirakenduse kasutajakogemust.

Kuna rakendus on tehtud, et kasutada seda puuteekraaniga nutitelefonis, kus kõik kasutajad on hästi harjunud, et kasutajaliides on ilusate animatsioonidega, oli vajadus leida raamistki, mis aitaks seda teha, kuna looda animatsioonid kasutades ainult JavaScripti on päris töömahukas ülesande. Selleks on üsna populaarne Reacti raamestki Framer Motion.

Esiteks pakub Framer Motion lihtsat ja intuitiivset süntaksit keerukate animatsioonide loomiseks, mida saab teha deklaratiivselt samas koodis kui komponente, millele neid kohaldatakse. See lihtsustab arendusprotsessi ja lihtsustab kõikide animatsioonide jälgimist, kuna need on integreeritud otse koodi.

Teiseks võimaldab Framer Motion suurt kontrolli animatsioonide üle. Animatsioone saab kohandada, määrates mitmesuguseid parameetreid, näiteks kergendust, kestust ja viivitust. See pakub laia valikut võimalusi unikaalsete animatsioonide loomiseks, mis sobivad rakenduse spetsiifilistele vajadustele.

Kolmandaks pakub Framer Motion jõudluse eeliseid, kasutades kaasaegsete brauserite uusimaid funktsioone. See kasutab animatsioonide renderdamiseks Web Animations API-d ja CSS-animatsioone, mis tagab, et need on sujuvad ja kiired isegi madalama tasemega seadmetes.

Lisaks toetab Framer Motion žestide ja interaktsioonide kasutamist, et luua kaasahaaravamaid ja interaktiivsemaid kogemusi. See hõlmab lohistamise, kerimise ja suumimiseks vajaliku nipsutamise toetust, mida saab animatsioonidesse integreerida, et luua keerulisemaid ja dünaamilisemaid interaktsioone.

Üldiselt pakub Framer Motion võimsat ja intuitiivset tööriistakomplekti Reacti veebirakenduste komponentide animeerimiseks, mis võib parandada kasutajakogemust ja rakenduse üldist väljanägemist. [15] [16]

3.3.5 Telegram BOT rakendus

Kui on vaja luua Telegram Bot, mis nõuab suurel hulgal API taotlusi, on mõistlik kasutada spetsiaalset teeke. See võimaldab mitte tegeleda API dokumentatsiooni lugemise ja API taotluse koostamisega. Selle asemel saab minna otse arendamise protsessi. Kõige populaarsemad raamistikud on tehtud Python keeles. Kolm nendest on python-telegram-bot, pyTelegramBotAPI ja AIOGram.

python-telegram-bot ja pyTelegramBotAPI on põhjalikud raamistikud, mis toetavad paljusid Telegram-boti funktsioone, sealhulgas sõnumite käitlemist ja „inline“ päringuid. Need raamistikud on hästi dokumenteeritud ja need on lihtne kasutada. Kuid need raamistikud on suhteliselt vanad, muudab see boti töö vähem efektiivseeks, mis võib põhjustada viivitusi boti reageerimisaegades.

Seevastu AIOGram on uuem teek, mis kasutab Pythoni asyncio, et pakkuda Telegrami botile suure jõudlusega blokeerimata I/O-d. See toetab kõiki Telegrami boti standardfunktsioone ja pakub ka täiustatud funktsioone, nagu „inline“ päringute käsitlemine, sõnumite redigeerimine ja boti klaviatuurid. Selle asünkroonne olemus võimaldab kiiremat reageerimisaega ja paremat skaleeritavust, mistõttu on see suurepärane valik suuremahuliste ja suure jõudlusega Telegram-botide loomiseks.

4 Rakenduse arendus

Selles osas tutvume rakenduse arendusprotsessiga, arutades erinevaid etappe. Uurime arhitektuurilist disaini, rakendamise üksikasju ja teel tekkinud probleeme.

4.1 Andmebaasi ERD mudeli loomine

Selleks, et rakenduses saaks seadistada koosoleku ja tema sündmused päris detailselt, on vaja võtta arvesse seda juba ERD mudeli loomisel.

- „Meeting“ olem – On peamine olem, mis esindab koosolekut. See olem on vanemolem järgmistele olemitele, nagu näiteks „Event“ olem, ehk sündmusolem. Tal on atribuudid, mis kirjeldavad koosoleku algust, lõppu ja maksimaalset eelarvet inimese kohta, kui see on vajalik. Samuti tal on seos „Meeting_User“ olemiga, mis näitab kes osaleb selles koosolekul.

- „Event“ olem – Esindab üht sündmust koosolekul. Ta sisaldab palju atribuuti, mis annavad võimalusi täpselt seadistada millal ja mis toimub ja kus see asub. Sel olemil on seos „Event_User“ olemiga, mis esindab kasutaja osalemist sellel sündmusel. Samuti on seos „Event_Navigation“ olemiga, mille abil on aru saadav sündmuste järjekord.

- „Requirement“ olem – Esindab sündmuse seotud nõudmisi. Ta sisaldab infot nõude kohta. Lisaks on seos „Requirement User“ olemiga, mis kirjeldab kes kavatseb kasutada seda nõuet.

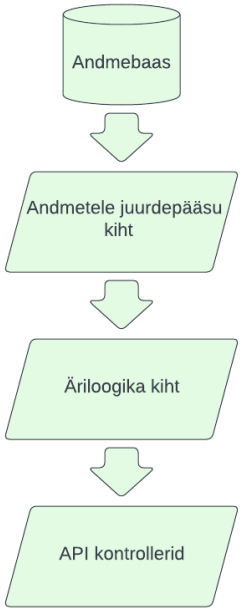
- „Requirement Parameter“ olem – Esindab sündmusega seotud nõudmiste täpsemad omadusi. See olem kirjeldab detailsem nõue omadused. Sel omadusel on seos „Requirement“ olemiga.

- „Payment“ olem – Esindab nõudmiste täitmisel tehtud makseid. Olem kirjeldab kes, millal ja kui palju maksis.

- „Money Transfer“ olem – Esindab raha üleandmist ühelt kasutajalt teistele. Kirjeldab millal ja kui palju raha üle kanti. Samuti sisaldab infot, kas anti üle sularahas või mitte.

4.2 Serverrakenduse loomine

Serverrakenduses on 3 põhi osa: andmetele juurdepääsu kiht, äriloogika kiht ja API kontrollerid. Andmetele juurdepääsu kiht suhtleb andmebaasiga päringute abil. Äriloogika kiht sisaldab ärireegleid, protsesse ja operatsioone, mis reguleerivad rakenduse käitumist. API kontrollerid omakorda annavad juurdepääsu äriloogika meetodile kasutades HTTP päringut.



Joonis 1. Lihtsustatud serverrakenduse skeem

4.2.1 Autoriseerimine

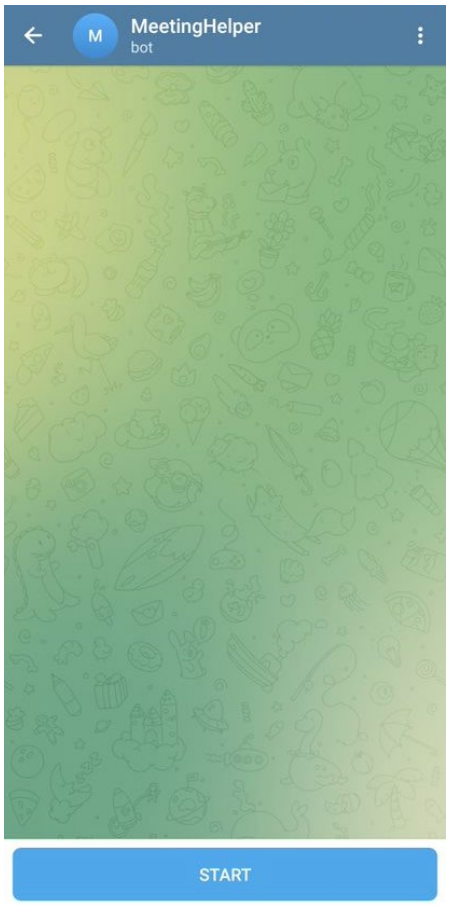
Autoriseerimiseks valiksin kasutada „Seamless Telegram Login“ kasutamise. „Seamless Telegram Login“ on Telegram messengeri pakutav funktsioon, mis võimaldab kasutajatel Telegrami kontot kasutades veebilehtedele või rakendustesse sisse logida. Seda nimetatakse "Seamless" ehk sujuv, sest kasutajad saavad hõlpsasti sisse logida, ilma et nad peaksid läbima eraldi registreerimisprotsessi või looma uut kasutajanime ja parooli.

Seamless Telegram Login'i abil saavad kasutajad kasutada oma olemasolevat Telegram-kontot, et autentida end veebilehes või rakenduses, mis muudab sisselogimise protsessi mugavamaks ja kiiremaks. See funktsioon tagab kasutajatele ka suurema turvalisuse, kuna nad ei pea jagama oma isiklikke andmeid ega looma uusi paroole iga veebisaidi jaoks, millele nad soovivad pääseda. Selle

asemel tegeleb autentimisega Telegram, mis kontrollib kasutaja identiteeti ja jagab tema andmeid turvaliselt veebisaidi või rakendusega.

Sisselogimise protsess algatatakse veebirakenduse avamisega, mis saab Telegramilt automaatselt kasutaja andmed, nagu Telegrami ID, täisnimi ja e-posti aadress (kui see on olemas). Nende andmete turvalisuse tagamiseks on siiski vaja kontrollida andmeid. Selleks kodeeritakse andmed SHA256-ga, kasutades võtmena täpselt seda bot API tokenit. Kui veebirakenduse genereeritud hash vastab Telegramilt edastatud hashile, kinnitab see, et andmed on turvalised ja on tõepoolest Telegramilt edastatud. See kontrolliprotsess aitab vältida volitamata juurdepääsu ja tagab kasutajaandmete autentsuse.

Lõplikul lahendusel see tähendab, et registreerimiseks kasutajal on lihtsalt vaja vajutada „START“ nuppu mis ilmub, kui esimene kord avad seda boti.



Joonis 2. Esialgne bot chati välimus

4.2.2 Võlgade kalkuleerimine

Võlgade arvutamine koosoleku ajal tekkinud kulude alusel, on üks selle rakenduse eeliseid. Kuid mõelda välja algoritm, mis pakub lahendada kõik võlud, kasutades võimalikult väheste tehingute arvu, polnud kõige lihtsam ülesanne. Seda võiks teha kui ka serverrakenduses, kui ka klientrakenduses. Kuid juhul, kui teha seda klientrakenduses tuleb välja palju nõrkusi. Esiteks see tähendab, et me peame andma juurdepääsu kõikidele tehingute andmetele, mis kindlasti pole väga turvaline. Teiseks see tähendab, et kliendi rakendus, mis peab töötama nii kiiresti, kui see on võimalik, peab tegelema lisa arvutamisega. Need pole liiga rasked arvutamised, aga kui on võimalik neid vältida tuleb seda teha. Ja viimaks, kui tulevikus tehakse alternatiivne kliendirakendus, siis tuleb see algoritm ümber teha. Sellepärast loodi algoritm, mis tagastab transaktsioon objektide list, oli loodud serverrakenduses äriloogika kihis ja sellele meetodile tehti juurdepääsu läbi API päringu.

Esiteks algoritm võtab kõik maksed, kulud ja muud rahatehingut ja salvestab nad sõnastiku, kus võti on kasutaja ja väärtus on maksete summa.

```
public async Task<IEnumerable<MoneyTransfer>> GetCloseDebtsTransfers(
    Guid meetingId,
    IRequirementService requirementService,
    IUserService userService,
    IPaymentService paymentService
)
{
    var personsExpenseInMeeting :Dictionary<Guid,double> = await GetPersonsExpenseInMeeting(
        meetingId,
        requirementService,
        userService,
        paymentService
    );
    var personsPaymentsInMeeting :Dictionary<Guid,double> = await GetPersonsPaymentsInMeeting(
        meetingId,
        requirementService,
        paymentService
    );
    var debts :Dictionary<Guid,double> = personsExpenseInMeeting.Merge(
        personsPaymentsInMeeting,
        mergeFunc: doubles :IEnumerable<double> => doubles.Sum()
    );
    return GetCloseDebtsTransfers(debts);
}
```

Joonis 3. Võlgade kalkuleerimise algoritm

Pärast, selle sõnastiku alusel tehakse makse tehingu objektid. Selleks, et vähendada tehingute arvu, on vaja leida kõige sarnasema suurusega võlgade ja maksete paar. 'Lähemad' tähendab pigem lähestikku paiknemist

```
public IEnumerable<MoneyTransfer> GetCloseDebtsTransfers(Dictionary<Guid, double> debts)
{
    var moneyTransfers = new List<MoneyTransfer>();

    var i = 0;
    while (i < 1000)
    {
        var moneyTransfer = CreateTransfer(debts);
        if (moneyTransfer != null)
        {
            moneyTransfers.Add(moneyTransfer);
        }
        else
        {
            break;
        }

        i++;
    }

    return moneyTransfers;
}
```

Joonis 4. Algoritm mis genereerib tehingu üleval kirjeldatud sõnastiku abil

```
private static MoneyTransfer? CreateTransfer(Dictionary<Guid, double> debts)
{
    foreach (var sender:Guid in debts.Keys.Where(user:Guid => debts[user] < 0))
    {
        var debtAmount:double = debts[sender];
        var receiver:Guid? = GetTransferReceiver(debts, sender);

        if (receiver == null || !(debtAmount < 0))
        {
            continue;
        }

        var deptAmount:double = debts[sender];
        var receiverRequire:double = debts[receiver.Value];
        var amount:double = receiverRequire >= Math.Abs(deptAmount) ? deptAmount : -1 * receiverRequire;

        debts[sender] -= amount;
        debts[receiver.Value] += amount;
        return new MoneyTransfer
        {
            Amount = Math.Abs(amount),
            SenderId = sender,
            ReceiverId = receiver.Value
        };
    }

    return null;
}
```

Joonis 5. Algoritm mis genereerib tehingu üleval kirjeldatud sõnastiku abil

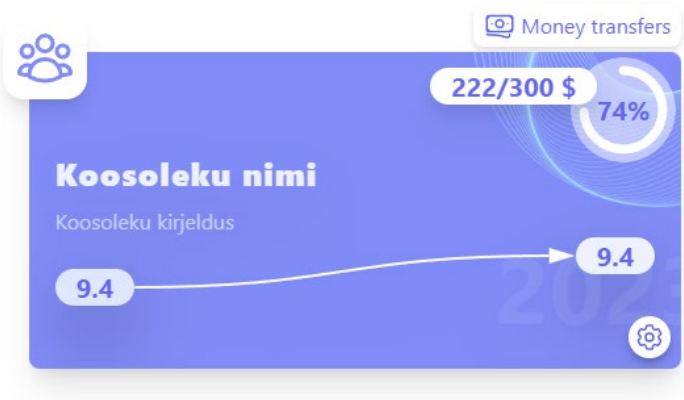
```
private static Guid? GetTransferReceiver(Dictionary<Guid, double> debts, Guid sender)
{
    var debtAmount:double = debts[sender];
    double? minDiff = null;
    Guid? minDiffReceiver = null;
    foreach (var receiver:Guid in debts.Keys.Where(user:Guid => user != sender && debts[user] > 0))
    {
        var diff:double = Math.Abs(debts[receiver] - debtAmount);
        if (minDiff == null)
        {
            minDiff = diff;
            minDiffReceiver = receiver;
        }
        else if (diff < minDiff)
        {
            minDiff = diff;
            minDiffReceiver = receiver;
        }
    }

    return minDiffReceiver;
}
```

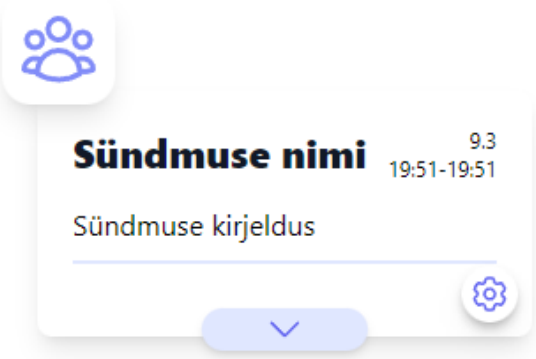
Joonis 6. Kõige lähemate võlgude ja maksete paari leidmise algoritm

4.3 UI komponendid

Kliendirakenduse-arendusprotsessi alguses loodi põhilised kasutajaliidese komponendid, sealhulgas kaardid sündmuste kohta info kuvamiseks, vormid selle teabe loomiseks ja muutmiseks, nupud erinevate tegevuste või navigeerimise jaoks, samuti dialoogid ja hüpikaknad. Selline lähenemisviis minimeeris võimalikke probleeme hilisemates arendusetappides ja võimaldas keskenduda andmete haldamisele serverrakendusest, loogika kirjutamisele ja lehe käitumise seadistamisele. See esialgne investeering kasutajaliidese komponentide kindla aluse loomisesse aitas tõhustada arendusprotsessi ja parandada lõpptoote üldist kvaliteeti.



Joonis 7. Koosoleku info kaart

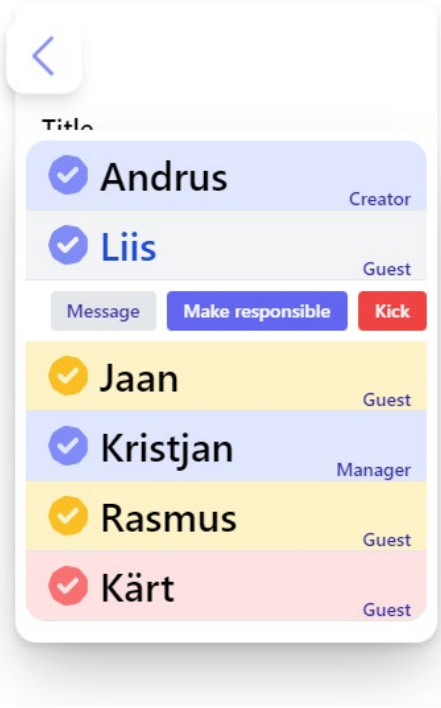


Joonis 8. Sündmuse info kaart

4.3.1 Külaliste nimekiri

Üks esimesi komponente, mis loodi, oli külaliste nimekiri. See komponent koosneb tegelikult kahest komponendist: nupust, mis avab ja sulgeb külaliste nimekirja, ja nimekirjast endast. Külaliste nimekirja komponent võimaldab kasutajatel vaadata kogu koosolekul või konkreetsel üritusel osalejate nimekirja. See funktsioon võimaldab kasutajatele lihtsat juurdepääsu olulisele teabele selle kohta, kes osalevad koosolekul või üritusel.

Külaliste nimekirja komponent ei võimalda mitte ainult koosolekust osavõtjate nimekirja vaadata, vaid võimaldab ka koosoleku korraldajatel teha nimekirja muudatusi. Näiteks saavad nad anda teatud külalistele privileege või neid koosolekult eemaldada. See funktsioon annab koosoleku korraldajatele suurema kontrolli ja paindlikkuse külaliste nimekirja haldamisel.

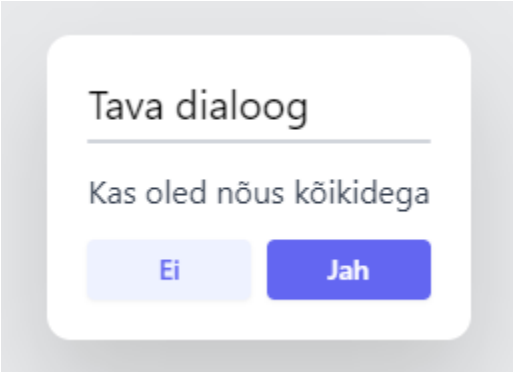


Joonis 9. Külaliste nimekiri (avatud)

4.3.2 Hüpikdialoog

Veebirakenduse arendusprotsessis loodi dialoogikomponent, et käsitleda tegevusi, mis nõuavad kasutajalt täiendavat kinnitust. Komponent loodi selleks, et vähendada koodi dubleerimist ja

lihtsustada dialoogide muutmist. Dialoogikomponent võimaldab muuta dialoogi pealkirja ja teksti, samuti dialoogis kuvatavate nuppude värvi ja tegevust. Selline lähenemine säästis arendusaega ja võimaldas suuremat paindlikkust dialoogide kujundamisel ja muutmisel kogu rakenduses.



Joonis 10. Hüpikdialooog komponent

4.3.3 Sündmuste ajaskaala

Veebirakenduse arendamisel oli üks kõige keerulisematest kasutajaliidese vaadetest sündmuste ajaskaala järjestus. Selle põhjuseks oli asjaolu, et rakendus toetab stsenaariumi, kus ühele sündmusele võib järgneda mitu sündmust. Noolte loomiseks, mis näitavad, milline sündmus järgneb teisele, kasutati raamistikku "react-archer". Raske ülesanne oli luua algoritm, mis genereerib sündmuste elemendid õiges järjekorras ja väldib sama sündmuse duplikaatide tekitamist.

Joonis 11. Sündmuste ajaskaala näide

4.3.4 Seadete menüü

Koosoleku muutmise kontekstis on vaja kasutajaliidese vahendeid, et teha vajalikke toiminguid. Seetõttu rakendati nende funktsioonide kuvamiseks kasutajaliidese komponent. See komponent sisaldab erinevaid tegevusnuppe, mida saab muuta erinevate kasutusjuhtumite jaoks, kusjuures sagedasemad toimingud on näiteks kustutamine ja muutmine. Vaatamata oma kompaktsele suurusele pakub see kasutajaliidese komponent märkimisväärset funktsionaalsust.

Joonis 12. Seadete menüü komponent (avatud ja suletud)

```
<EditMenu items={[
    { icon: "add", action: () => {console.log("add action")} },
    { icon: "edit", action: () => {console.log("edit action")} },
    { icon: "delete", action: () => {console.log("delete action")} }
]}/>
```

Joonis 13. Seadete menüü komponendi kasutusnäide

4.4 Päringut serverrakendusest

Kliendi rakenduse töö ajal tehakse üsna palju HTTP päringut serverrakendusele. Kõik saatud ja saadut info saadakse JSON formaadis, mis võimaldab seda lihtsalt konverteerida objektile, millega programm pärast tegelakse. Selleks et serverrakendus teadis, kes küsib andmed ja mis andmed saab näidata selle kasutajale, on vaja looda mingi autoriseerimise protsess. Selleks rakenduses selleks kasutakse kaks pealkiri HTTP päringus, ehk Header väärtused. Üks nendes on Telegrammi kasutaja andmed ja nende andmete kodeeritud hash. Need mõlemad väärtused klient rakendus saab Telegramist, kui veebileht on avatud WebApp'ina.

4.5 Telegram boti loomine

Telegram BOTi arendamise alguses oli esimene samm luua boti konto, kasutades Telegrami ametlikku boti "Bot father" ja saada bot API token. Seejärel seadistada botile tehtud kliendirakendus. Kui esialgne seadistamine oli lõpule viidud, kasutati Aiogram raamatukogu, et rakendada bot'i põhilist käitumist, mis võis vastu võtta sõnumeid ja muud meediasisu, mis saadeti

bot'ile. Veel üks põhi vajadus oli saata päringut serverrakendusele. Aga erinev asi on see, et autoriseerimiseks, on vaja eraldi saata kasutaja põhi andmed ja samade andmete kodeeritut hash,

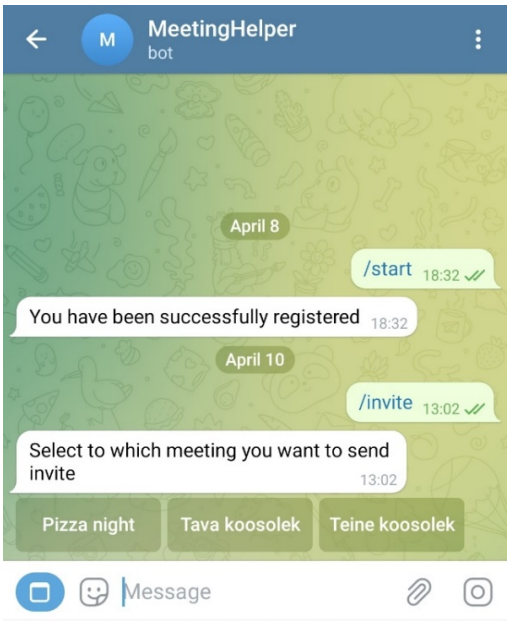
Arenduse jätkudes rakendati keerulisemaid sõnumikäsitlejaid, et võimaldada kasutajatel saata kutsed kohtumistele, kasutades oma Telegrami kontaktide nimekirja. Samuti lisati võimalus kasutajatel kutsete vastuvõtmist või tagasilükkamist ning meeldetuletuste saamist eelseisvate sündmuste või muude teadete saatmine.

Üldiselt osutus Telegram BOTi kasutamine koos Aiogram'i raamestikuga väga tõhusaks lahenduseks, et luua platvorm, mis suudab suhelda kasutajatega reaalajas, võimaldades pakkuda koheseid teateid ja uuendusi seoses kavandatud sündmuste ja kohtumistega.

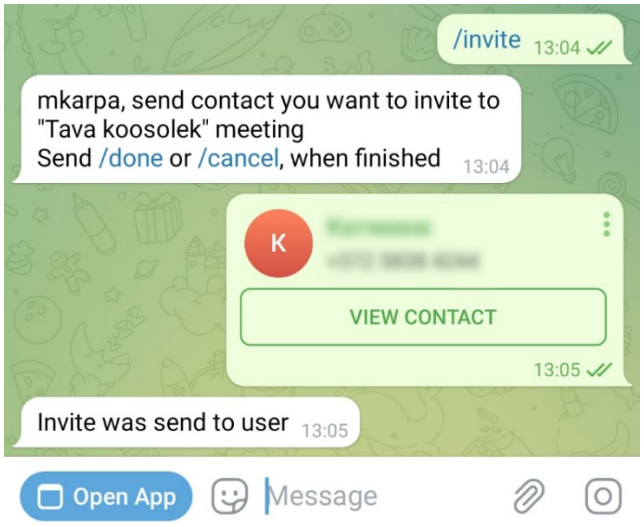
4.6 Kutsete saatmine

Selleks, et võimaldada koosolekukutsete saatmist Telegram BOTi kaudu, kasutati mitmeid Telegrami vahendeid, sealhulgas „Inline Keyboard Markup“, bot seisundid ja sõnumikäsitleja filtreerimine. Protsess ise hõlmab kasutaja järgmisi samme:

- "/invite" käsu saatmine
- Koosoleku valimine, kuhu kutse tuleks saata
- Kõigi soovitud kontaktide saatmine, Telegrami kontaktide nimekirjast



Joonis 14. Koosoleku valimine kutsumisprotsessis



Joonis 15. Kontakti saatmine kutsumisprotsessis

5 Tehtud rakenduse hinnang

Lõputöös esitatakse koosolekute korraldamise abirakendus, mis on integreeritud Telegramiga kui veebirakendusega. Rakenduse arendamisel kasutati kaasaegseid tehnoloogiaid nagu ASP.NET, Entity Framework, React ja Tailwind CSS. Arendusprotsessi alguses läbi viidud analüüs andis hea ülevaate planeeritud funktsionaalsustest ja võimalikest viisidest nende rakendamiseks. Rakenduse disain ja kasutajakogemuse kujundamine võimaldas tõhusamat arendusprotsessi. Saadud rakendus pakub üksikasjalikku ja paindlikku sündmuste seadistamist, mis ületab konkureerivate rakenduste funktsionaalsust. Kuigi kõiki kavandatud funktsionaalsusi ei suudetud täita, annab rakendus tugeva aluse edasiseks arendamiseks ja sellel on selged võimalused laiendamiseks. Kokkuvõttes võib projekti pidada edukaks, kuna see vastab esitatud nõuetele ja näitab kaasaegsete veebiarendustehnoloogiate võimalusi.

5.1 Võimalused edasi arenduseks

Lisaks sellele võib kolmandate osapoolte teenuste, näiteks kalenderrakenduste ja asukohateenuste integreerimine parandada kasutajakogemust ja pakkuda terviklikumat lahendust koosolekute korraldamiseks. Lisaks sellele võib teavituste ja meeldetuletuste süsteemi täiustamine suurendada kasutajate kaasatust ja osavõtumäära.

Lisaks toote kvaliteedi parandamisele saab rakendust edasi arendada, et lisada küsimustike täiustatud funktsionaalsus. See aitaks kasutajatel teha otsuseid, mis sobivad kõige paremini kõigile osalejatele. Teine võimalik funktsioon, mida võiks lisada, on võimalus luua avalikke sündmusi, mis on kõigile huvilistele nähtavad. See võimaldaks kasutajatel mitte ainult planeerida kohtumisi oma olemasolevate sõpradega, vaid ka laiendada oma suhtlusringkondi ja kohtuda uute inimestega.

6 Kokkuvõte

Käesoleva lõputöö eesmärk oli töötada välja uus abirakendus koosolekute korraldamiseks, kasutades Telegrami integratsiooni. Lahendus sisaldab sujuvat autoriseerimist, kasutades Telegrami tehnoloogiaid, koosoleku loomist ja haldamist ning soovitatud tehinguid, et anda kõik võlad vastavalt koosoleku ajal tehtud tehingutele. Olemasolevate lahenduste analüüsimisel tuvastati nende nõrkused ning arendusprotsessis kasutati kaasaegseid tehnoloogiaid, nagu ASP.NET ja Entity Framework, React ja Tailwind CSS. Kuigi mõned funktsionaalsused jäid töömahu tõttu lõpetamata, pakub rakendus enamikku täpsustatud nõuetest ja omab tugevat alust edasiseks arendamiseks. Üldiselt võib projekti pidada edukaks, kuna see täidab olemasolevate lahenduste lüngad ja pakub tõhusamat viisi koosolekute korraldamiseks.

Kasutatud kirjandus

[1] J. Duffy, "pcmag.com," ZIFF DAVIS, LLC, 2 Mai 2017. [Online]. Available: https://www.pcmag.com/reviews/doodle. [Accessed 5 Aprill 2023].

[2] S. Fisher, "Lifewire.com," Dotdash Meredith, 17 August 2022. [Online]. Available: https://www.lifewire.com/google-calendar-review-1357929. [Accessed 5 Aprill 2023].

[3] P. Singh, "Mobile app daily," MobileAppDaily, 27 Märts 2023. [Online]. Available: https://www.mobileappdaily.com/app-review/meetup-app. [Accessed 5 Aprill 2023].

[4] D. Nield, "Popular Science," 8 Veebruar 2018. [Online]. Available: https://www.popsci.com/event-organizing-apps/. [Accessed 5 Aprill 2023].

[5] C. P. Nate Drake, "techradar.pro," Future US Inc, 3 Veebruar 2022. [Online]. Available: https://www.techradar.com/reviews/trello. [Accessed 5 Aprill 2023].

[6] J. Duffy, "pcmag.com," ZIFF DAVIS, LLC, 13 Juuli 2021. [Online]. Available: https://www.pcmag.com/reviews/trello. [Accessed 5 Aprill 2023].

[7] D. Tobin, "Integrate.io," Integrate.io, 1 Märts 2023. [Online]. Available: https://www.integrate.io/blog/which-database/. [Accessed 7 Arill 2023].

[8] The Upwork Team, "upwork," 16 Juuli 2021. [Online]. Available: https://www.upwork.com/resources/ms-sql-vs-mysql-which-relational-database-is-right-for-you. [Accessed 7 Aprill 2023].

[9] K. UBAH, "Make use of," Valnet Inc, 6 Aprill 2023. [Online]. Available: https://www.makeuseof.com/most-popular-backend-frameworks/. [Accessed 9 Aprill 2023].

[10] P. Prema, "Medium," 4 Oktoober 2021. [Online]. Available: https://medium.com/@putuprema/spring-boot-vs-asp-net-core-a-showdown-1d38b89c6c2d. [Accessed 9 Aprill 2023].

[11] L. d. Alba, "Sitepoint," 9 Märts 2023. [Online]. Available: https://www.sitepoint.com/vue-vs-react/. [Accessed 9 Aprill 2023].

[12] Simplilearn, "Simplilearn," 11 November 2021. [Online]. Available: https://www.simplilearn.com/react-vs-angular-vs-vue-article. [Accessed 9 Aprill 2023].

[13] Abdu, "Neoito," 10 Märts 2021. [Online]. Available: https://www.neoito.com/blog/angular-vs-react-vs-vue/. [Accessed 9 Aprill 2023].

[14] T. White, "Dev.to," 23 Juuli 2021. [Online]. Available: https://dev.to/turpp/what-is-tailwind-css-with-react-js-45ng. [Accessed 10 Aprill 2023].

[15] G. Lewington, "logrocket.com," 29 Juuli 2021. [Online]. Available: https://blog.logrocket.com/framer-motion-tutorial/. [Accessed 10 Aprill 2023].

[16] T. Victory, "Openreplay.com," 3 Jaanuar 2023. [Online]. Available: https://blog.openreplay.com/doing-animations-in-react-with-framer-motion/. [Accessed 10 Aprill 2023].

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

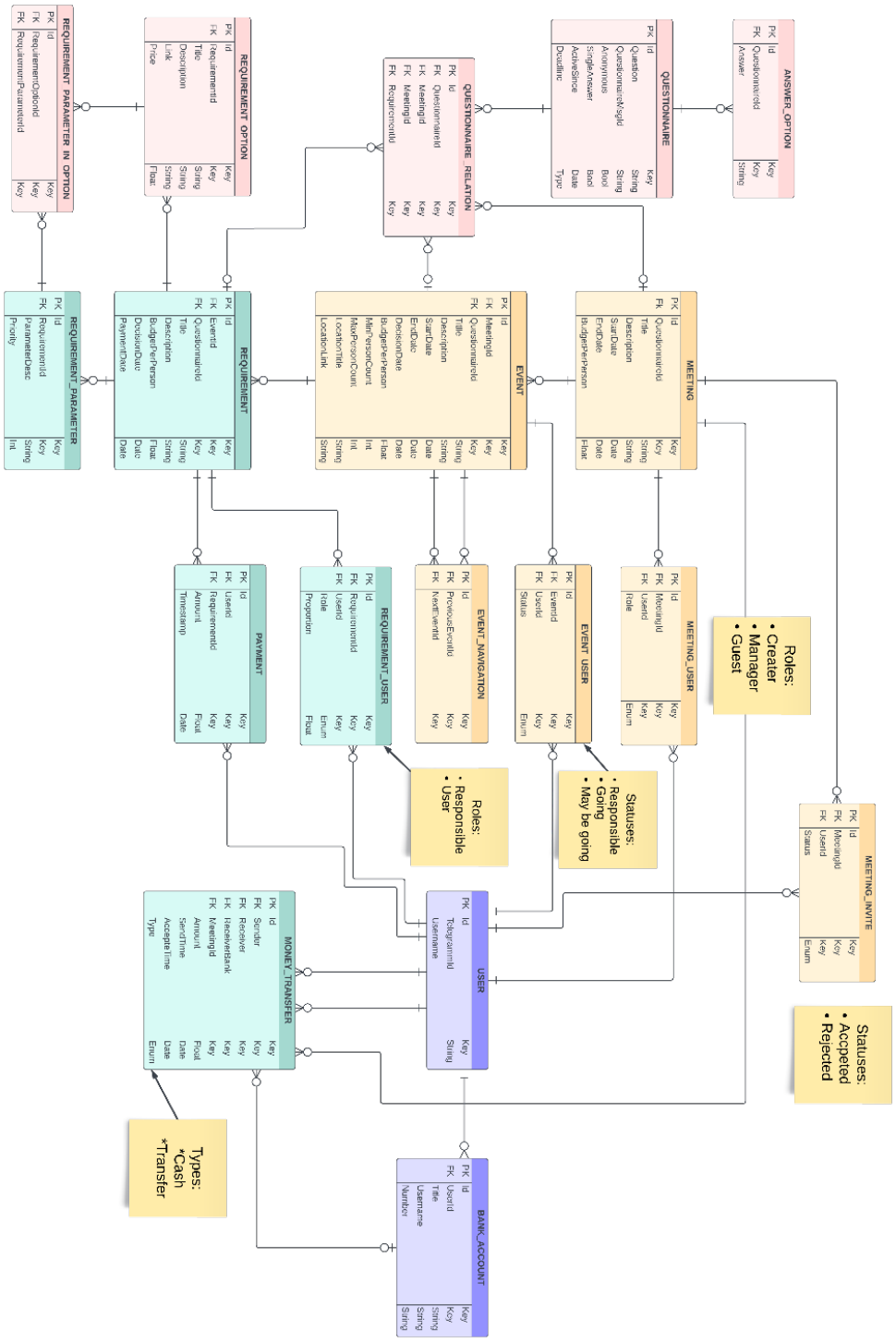
Mina, Marko Karpats

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

24.04.2023

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 – Adnmebaasi ERD mudel



Joonis 16. Andmebaasi ERD mudel