TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Liis-Marie Kütt 185359IAIB

Algkooliõpilaste rahaliste vahendite haldamise rakenduse arendus

Bakalaureusetöö

Juhendaja: Martin Verrev
MSc

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: Liis-Marie Kütt

18.05.2021

Annotatsioon

Oma rahaliste vahendite haldamise, planeerimise ja finantsotsuste eest vastutamise oskused on noore inimese arengus äärmiselt olulised ja seepärast on tähtis alustada nende omandamist võimalikult vara. Olemasolevad rahaliste vahendite haldamise lahendused on algkooliõpilastele kasutamiseks liiga keerulised või piiratud kasutustingimustega, nende kasutamiseks peab olema täisealine.

Bakalaureusetöö eesmärk oli arendada algkooliõpilaste rahaliste vahendite haldamise rakendus, mis võimaldaks administreerida sissetulekuid ja väljaminekuid, püstitada kogumiseesmärke ja jälgida nende täitmist.

Töö tulemusel valmis algkooliõpilastele mõeldud keskkond https://hoiuporsas.com/, mis on kasutatav hübriidrakendusena nii töölaual kui mobiiltelefonis. Töö teoreetiline osa kirjeldab valminud rakenduse arendusprotsessi.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 35 leheküljel, 6 peatükki, 12 joonist, 0 tabelit.

Abstract

Development of a funds management application for primary school pupils

Skills in managing one's own funds as well as planning and taking responsibility for financial decisions are crucial for a young person's development and it is therefore important to start acquiring them as early as possible. Existing fund management solutions are too complex for primary school pupils to use or have limited conditions of use, the users must be of legal age.

The aim of the Bachelor's thesis was to develop an application for the management of funds of primary school pupils, which would enable the administration of income and expenses, the setting of savings goals and the monitoring of their fulfilment.

As a result of the work, an environment for primary school students was created at https://hoiuporsas.com/. This solution can be used as a hybrid application on both desktop and mobile phones. The theoretical part of the work describes the development process of the completed application.

The Bachelor’s thesis is written in Estonian and contains 35 text pages, 6 chapters, 12 figures, 0 tables.

Lühendite ja mõistete sõnastik

API	*Application Programming Interface*, liides serveriga suhtlemiseks
boilerplate	Koodilõigud, mida korratakse mitmes kohas muutmata kujul või väikeste muudatustega
builder	Objektide loomismuster, mis võimaldab keerukaid objekte samm-sammult konstrueerida
clean code	Kood, mida on lihtne mõista ja mida on lihtne muuta
CLI	*Command-Line Interface*, töötleb arvutiprogrammile käske tekstiridade kujul
commit	Versioonijuhtimissüsteemides toiming, mis saadab lähtekoodi viimased muudatused hoidlasse, tehes need muudatused hoidla peaversiooni osaks
CORS	*Cross-Origin Resource Sharing*, brauserimehhanism, mis võimaldab kontrollitud juurdepääsu väljaspool antud domeeni asuvatele ressurssidele
CSS	*Cascading Style Sheets*, märgistuskeel, mida kasutatakse peamiselt veebilehe kujundamisel
DevOps	Praktika, mis ühendab tarkvaraarenduse ja IT-toimingud
eesrakendus	*Front-end*, andmete konverteerimine graafiliseks liideseks HTML, CSS ja JavaScripti abil, et kasutajad saaksid andmeid vaadata ja nendega suhelda
endpoint	Sidekanali asukoht, kust API-d saavad juurdepääsu ressurssidele, mida nad oma funktsiooni täitmiseks vajavad
event	Tarkvara poolt tunnustatud toiming või juhtum
event handling	Rutiin, mis tegeleb sündmusega, võimaldades programmeerijal kirjutada sündmuse toimumisel käivitatavat koodi
frontend first	Eesrakendust arendatakse enne tagarakenduse arendamist
full-stack	Kogu arvutisüsteem või rakendus, mis hõlmab nii ees- kui ka tagarakendust
getter	Javas kasutatav meetod, mis loeb muutuja väärtust ja tagastab selle
HTTP	*Hypertext Transfer Protocol*, rakenduskihi protokoll hüpermeedia dokumentide edastamiseks

IDE	*Integrated Development Environment*, tarkvararakendus, mis pakub programmeerijatele tarkvaraarenduseks terviklikke võimalusi
issue	Kirja pandud üksik töö, mis on vaja lõpetada
JDK	*Java Development Kit*, arenduskeskkond Java programmeerimiskeelt kasutavate rakenduste ja komponentide loomiseks
JSON	*JavaScript Object Notation*, andmevahetusformaat, mis hoiab andmeid nime väärtus paaridena
JWT	*JSON Web Token*, JSON-i põhine ligipääsulubade standard
lingua franca	Keel või keelte segu, mida kasutavad ühise suhtlusvahendina erineva emakeelega inimesed
milestone	Kasutatakse *issue*-de kindlaksmääratud gruppi koondamiseks
prop	Viis, kuidas komponendid saavad andmeid vanemkomponentidelt vastu võtta
proxy	Serverirakendus või seade, mis toimib vahendajana klientide päringutele
PWA	*Progressive Web Application*, rakendustarkvara, mis on mõeldud töötama standarditele vastavat brauserit kasutaval mis tahes platvormil
REST	*Representational State Transfer*, tarkvara arhitektuuristiil, mis kasutab HTTP protokolli
REST API	API, mis vastab arhitektuuristiili REST piirangutele ja võimaldab suhelda veebiteenustega
setter	Javas kasutatav meetod, mis muudab muutuja väärtust
sõrestikmudel	*Wireframe*, madala- ja keskmise detailsusastmega prototüüp
SQL	*Structured Query Language*, standardkeel andmete salvestamiseks, modifitseerimiseks ja andmebaasist kättesaamiseks
SSL	*Secure Sockets Layer*, protokoll autenditud ja krüptitud linkide loomiseks võrgustatud arvutite vahel
tagarakendus	*Back-end*, andmetele juurdepääsukiht, mis üldjuhul tegeleb eesrakenduse tehtud päringutega
template literal	Mall, mille korral on komponendi HTML ja JavaScript samas failis
URI	*Uniform Resource Identifier*, ainulaadne tähemärkide jada, mis tähistab veebitehnoloogiate poolt kasutatava loogilist või füüsilist ressurssi

URL	*Uniform Resource Locator*, selline viide veebiressursile, mis määrab ressursi asukoha arvutivõrgus ja selle hankimise mehhanismi
web dyno	Virtuaalse masina instants
ZIP	Arhiivifailivorming, mis toetab kadudeta andmete kokkupakkimist

Sisukord

Jooniste loetelu

1 Sissejuhatus

Käesoleva bakalaureusetöö teema on rakenduse arendus, mis võimaldab hallata rahalisi vahendeid ja panna kirja konkreetseid kogumiseesmärke.

Hetkel puudub algkooliõpilastele mõeldud lahendus nende rahalise seisu haldamiseks ja kogumiseesmärkide kirja panekuks. Eksisteerib platvorme, kus on võimalik jälgida oma rahalist seisu, aga need on sihtgrupi jaoks liiga keerulised ning arusaamatud.

Töö eesmärk on arendada rakendus, mida algkooliõpilased saavad kasutada oma sissetulekute ja väljaminekute registreerimiseks ning raha kogumise eesmärgistamiseks. Rakendus on mõeldud noortele vanuses 7-12, seega peab see olema võimalikult lihtne ja arusaadav.

Loodava rakenduse eesmärk on arendada lastes vastutustunnet ning aidata neil silma peal hoida oma rahalisel seisul, õpetades rahaga ümberkäimist ning säästmist ja sellega seonduvalt planeerivat ning vastutustundlikku käitumist. Kogumiseesmärkide üles märkimine aitab hoida sihti silme ees ja seeläbi suunata noort oste paremini läbi mõtlema ning planeerima.

Lõputöö õnnestumiseks viidi läbi vestlused potentsiaalsete kasutajatega ning analüüsiti analoogseid olemasolevaid rakendusi. Järgmise sammuna töötati välja rakenduse disain ja loodi prototüüp Figmas, mida valideeriti kasutajate peal.

Rakenduse teenuskiht programmeeriti keeles Java, kasutades Spring Boot raamistikku. Vastavalt Figma prototüübile loodi esitluskiht, kasutades Vue.js raamistikku. Lisaks on rakenduse üldise väljanägemise muutmiseks kasutatud märgistuskeele CSS (*Cascading Style Sheets*) võimalusi. Andmebaas realiseeriti, kasutades PostgreSQL. Valminud rakenduse adekvaatsuse hindamiseks anti see kasutajatele testimiseks ning hinnati ka rakenduse vastavust nõuetele.

Töö järgnevates osades antakse detailsem ülevaade projekti teostamisest. Analüüsitakse olemasolevaid rakendusi ja käsitletakse rakendusele seatud nõuded. Kirjeldatakse erinevate tehnoloogiate kasutamist, arenduskeskkonna seadistamisprotsessi ja

tööprotsessi ülesehitust. Samuti selgitatakse kasutatud arhitektuuri, disaini ja loodud koodi põhimõtteid. Lõpetuseks analüüsitakse valminud rakenduse vastavust seatud nõuetele, tuuakse välja kasutajatestimise tulemused ning arutatakse edasiarendamise võimalusi tulevikus.

2 Analüüs

Selles peatükis antakse ülevaade olemasolevatest alternatiivsetest lahendustest ja rakenduse nõuetest.

Hetkel puudub algkooliõpilastele mõeldud lahendus rahalise seisu jälgimiseks ja kogumiseesmärkide kirjapanekuks. Eksisteerib erinevaid platvorme, kus on võimalik jälgida oma rahalist seisu, kuid need on algkooliõpilastele raskesti hallatavad.

2.1 Alternatiivsed lahendused

Järgnevalt tutvustatakse kolme olemasolevat rakendust, mille abil on võimalik oma rahalist seisu registreerida. Lahenduste analüüsimisel keskendutakse peamiselt tehingutega opereerimisele ning eesmärgistamisele, sest need on loodavas rakenduses fookuses. Alternatiivsete rakenduste analüüsimisel kasutati Google Chrome veebibrauseri versiooni 90.0.4430.85.

Swedbank Rahaplaneerija ja SEB Rahapäevik rakenduste kasutamiseks peab olema vastava panga klient, seetõttu jäid need antud analüüsist välja.

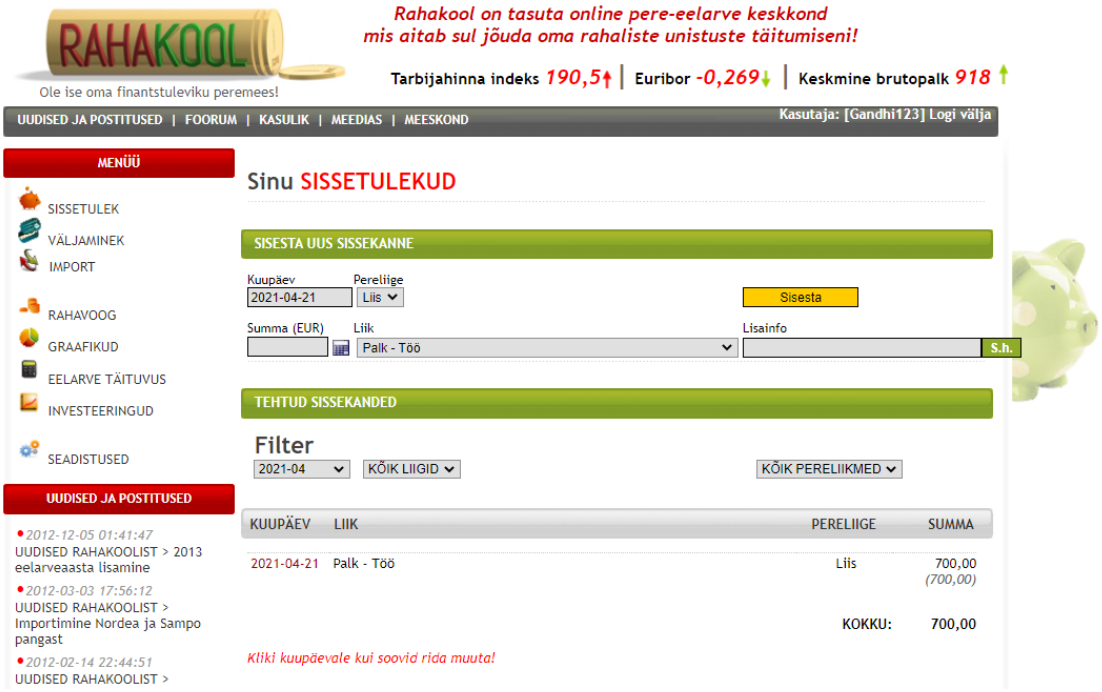
2.1.1 Rahakool

Rahakool võimaldab kasutajal sisestada pereliikmeid, seadistada enda tulu- ja kululiigid ning kategooriad. Rakenduses saab planeerida igale tulu- ja kululiigile soovitud aastase eelarve, sisestada sissetulekuid ja väljaminekuid, vaadata perioodi rahavoogu ning planeeritud eelarvet võrdluses tegelike näitajatega. [1]

Pärast rakendusse sisselogimist tuleb avalehel ette tegevuste valikuvõimalus ning juhend tehingute sisestamiseks (Joonis 1). Selleks, et kasutaja saaks sissetulekuid ja väljaminekuid kirja panna, peab süsteemi sisestama oma pereliikmed, tulu- ja kululiigid ning kategooriad. Minnes enne nende tegevuste lõpuleviimist lisama tehingut, annab süsteem küll teada, et tehing lisati, kuid tegelikult seda ei tehtud.

Joonis 1. Kuvatõmmis Rahakooli sisse loginud kasutaja avalehele vaatest.

Sissekande sisestamisel tuleb kõigepealt minna sissetulekute või väljaminekute lehele. Lehe päises on uue kande loomise vorm ja selle all on võimalik vaadata sissetulekuid või väljaminekuid ning neid filtreerida kuude, liigi ja pereliikme järgi (Joonis 2). Sissekande lisamisel tuleb määrata kuupäev, pereliige, summa ja tehingu liik. Soovi korral saab kirjutada ka lisainformatsiooni tehingu kohta.



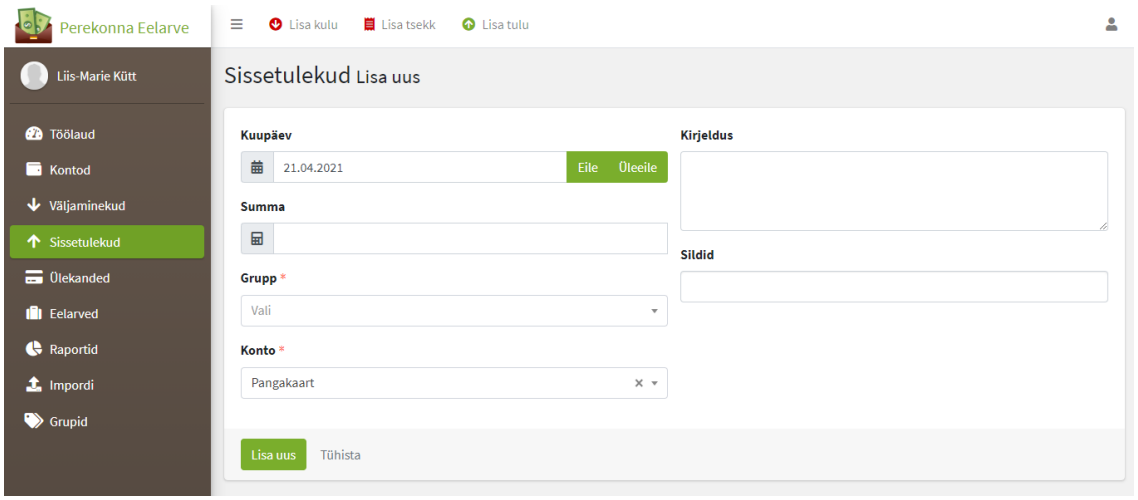
Joonis 2. Kuvatõmmis Rahakooli sissetulekute vaatest.

Rahakooli veebilehel on võimalik seadistada eelarvet, kuid konkreetsemaid kogumiseesmärke seada ei saa. Kasutaja saab rahavoogu vaadata kuude kaupa, muid perioode ei ole võimalik otsingusse sisestada.

2.1.2 Perekonna Eelarve

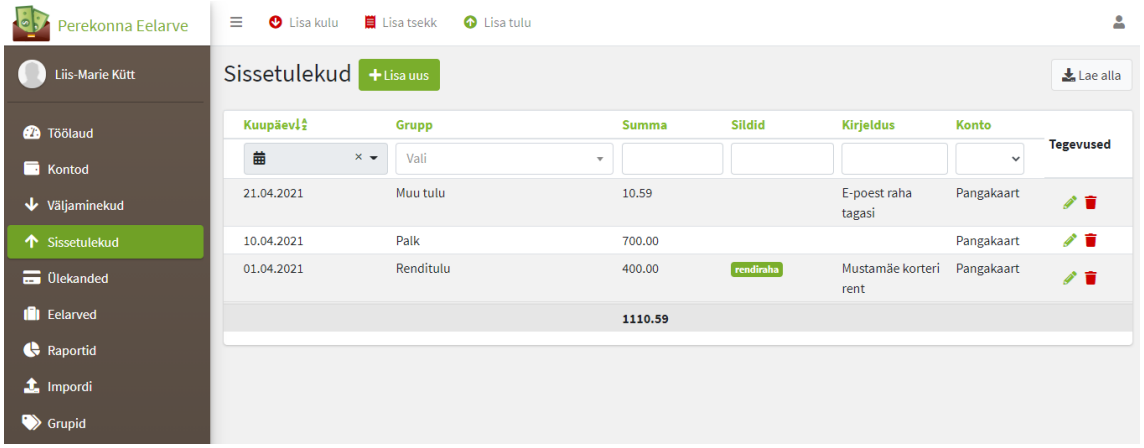
Perekonna Eelarve näol on tegemist veebilehega, mis võimaldab kasutajal sisestada oma sissetulekuid ja väljaminekuid. Rakendus reklaamib ka eesmärkide seadmise võimekust. Samuti saab kasutaja importida pankade CSV faile, eksportida andmeid Excelisse, koostada raporteid ning luua eelarvet ja seda pereliikmetega jagada. [2]

Sissetuleku ja väljamineku lisamise vormid on realiseeritud eraldi lehtedena (Joonis 3). Tehingule tuleb lisada kuupäev, summa, grupp ja konto. Lisaks on tehingule võimalik juurde märkida ka täpsem kirjeldus ning lisada silte. Kui üritada lisada tehingut ilma summat märkimata, siis tehingu lisamine ei õnnestu, kuid veateadet kasutajale ei kuvata ja täidetud vorm jääb lahti. Kui tehingule silte ei lisata, lisab süsteem ühe ilma tekstita tühja sildi. Sama vormi kasutatakse, et kuvada tehingu detailvaadet, kuid seal puudub kustutamisnupp.

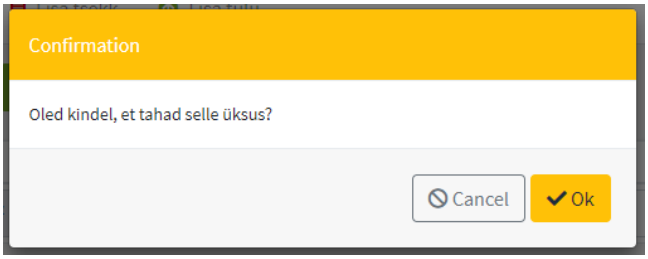


Joonis 3. Kuvatõmmis Perekonna Eelarve sissetuleku lisamise vaatest.

Sissetulekute ja väljaminekute vaatamiseks on eraldiseisvad lehed. Kirjeldus on sissetulekute koondlehe põhjal (Joonis 4). Sissetulekuid on võimalik sorteerida ja filtreerida järgnevate atribuutide järgi: kuupäev, grupp, summa, sildid, kirjeldus ja konto. Kustutamisel tuleb ette kinnitusmodaal, mille tekst on segane ja selles on kasutatud läbisegi nii eesti kui ka inglise keelt (Joonis 5).



Joonis 4. Kuvatõmmis Perekonna Eelarve sissetulekute vaatest.



Joonis 5. Kuvatõmmis Perekonna Eelarve tehingu kustutamise kinnitus modaalist.

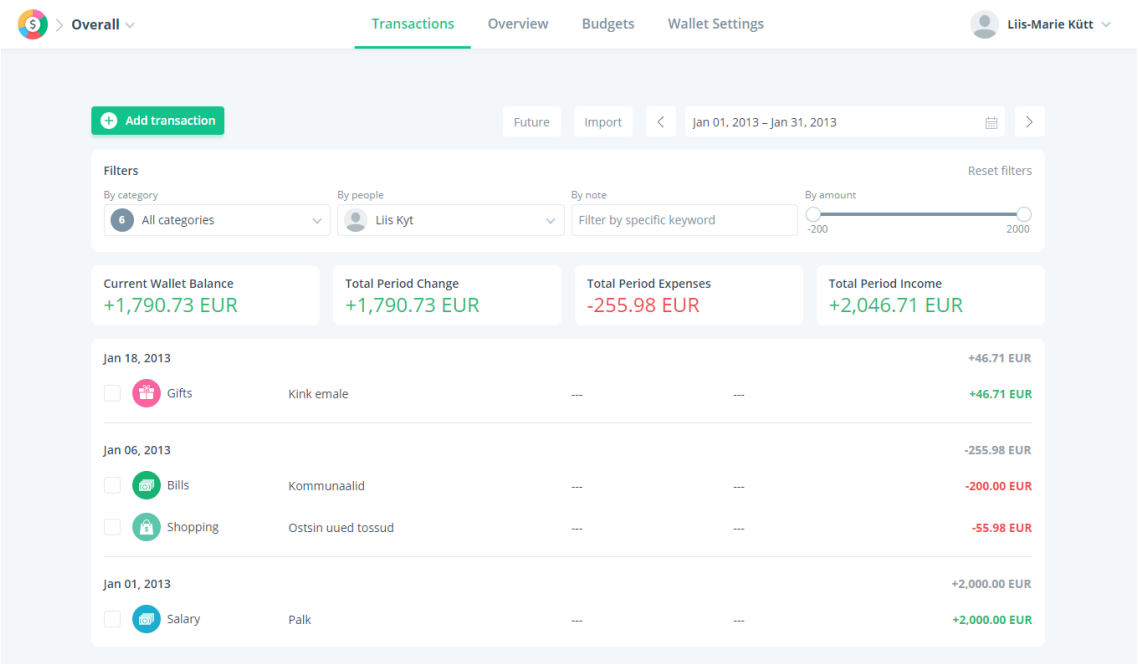
Perekonna Eelarve rakenduses saab seadistada eelarveid, kuid konkreetsemaid kogumiseesmärke seada ei ole võimalik.

2.1.3 Spendee

Spendee pakub võimalust hallata oma rahalisi vahendeid. Rakenduses saab luua jagatud rahakotte, isikupärastada kategooriaid, kasutada erinevaid valuutasid ja luua meeldetuletusi ning teavitusi. Spendee'ga on võimalik siduda ka pangakontot. [3]

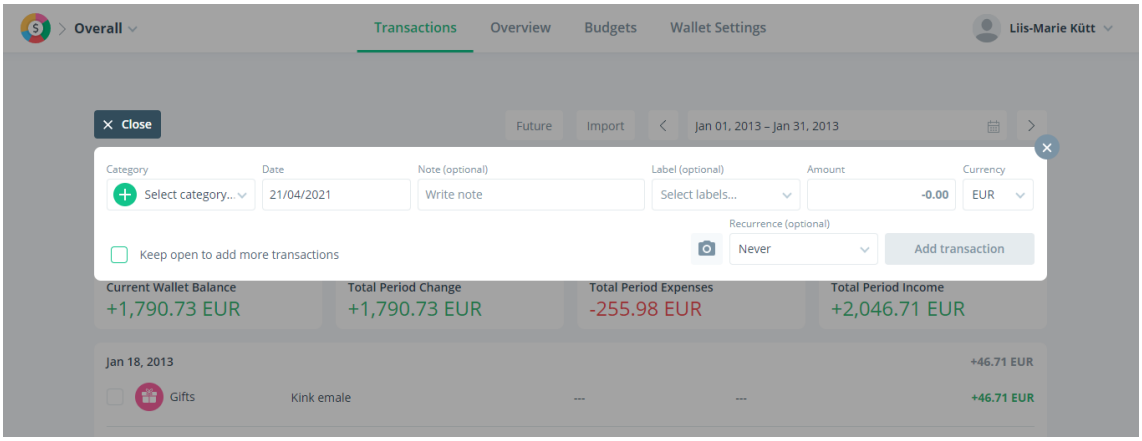
Spendee rakenduse kasutamine on lihtne ja loogiline, kuid sellel puudub eesti keele tugi. Noorematel kui 15-aastastel ei ole lubatud rakendusse kasutajakontot teha [4].

Tehingute vaates saab kasutaja jälgida oma tehinguid ja neid vastavalt soovile filtreerida (Joonis 6). Tehinguid on võimalik filtreerida kategooria, inimeste, kirjelduse ja summa järgi. Kasutajal on võimalus muuta ja käsitsi sisestada vaatluse all olevat perioodi. Tehingute loendvaade on kompaktne ja kergesti hallatav.



Joonis 6. Kuvatõmmis Spendee tehingute vaatest.

Tehinguid saab lisada (Joonis 7) ja muuta otse tehingute vaates avanevas modaalis. Mõlemal juhul on kasutatud sama komponenti, aga tehingu detailvaate juurde on lisatud kustutamisnupp. Lisamisnupp ei ole aktiivne, kui kohustuslikud väljad on täitmata.



Joonis 7. Kuvatõmmis Spendee tehingu lisamise modaalist.

Spendee rakenduses saab seadistada eelarveid, kuid konkreetsemaid kogumiseesmärke ei ole võimalik seada.

2.1.4 Alternatiivsete lahenduste analüüs

Eespool nimetatud rakendused on pigem suunatud täiskasvanutele ja seetõttu sisaldavad funktsionaalsusi, mida algkooliõpilastel ei ole vaja. Samuti on need rakendused sihtgrupi jaoks liiga keerulised ja arusaamatud. Lisaks analüüsitud lahendustele on olemas veel mitmeid alternatiivseid rakendusi, kuid paljudel neist on defineeritud kasutaja vanusepiirang, mille tõttu ei ole algkooliõpilastel võimalik neid kasutada.

Rahakooli, Perekonna Eelarve ja Spendee analüüsi tulemused, kasutades Jakob Nielseni 10 kasutatavuse heuristikat kasutajaliidese kujundamisel [5] on välja toodud Lisas 1.

2.2 Nõuded rakendusele

Käesolevas peatükis tuuakse välja rakenduse peamised funktsionaalsed ja mittefunktsionaalsed nõuded. Nõuded koostati, võttes arvesse konkurentide analüüsi, intervjuusid sihtgruppi kuuluvate noortega ja autori enda nägemust. Esialgsete vajaduste visualiseerimiseks kujundati vaated, kasutades rakendust Figma. Sõrestikmudelid rakenduse esialgsest versioonist on välja toodud Lisas 2.

2.2.1 Funktsionaalsed nõuded

Käesolevas peatükis käsitletakse funktsionaalsed nõuded, millele rakendus peab vastama.

Rakendusele seatud funktsionaalsed nõuded on järgmised:

- Rakenduses on võimalik luua kasutajakontot.
- Rakendusse on võimalik logida sisse olemasoleva kasutajana.
- Kasutaja saab vaadata oma profiili.
- Kasutaja saab välja logida.
- Kasutaja näeb ülevaadet oma sissetulekutest ja väljaminekutest.
- Kasutajale kuvatakse sissetulekuid ja väljaminekuid kuupäeva järgi kahanevas järjekorras. Loendvaates kuvatakse ka iga päeva rahalist kokkuvõtet.
- Kasutajale kuvatakse sissetulekuid ja väljaminekuid nädala, kuu, kõik korraga ja kasutaja enda valitud perioodi kaupa, sealjuures kuvatakse ka perioodi rahalist kokkuvõtet.
- Kasutaja saab sissetulekuid ja väljaminekuid otsida kategooria ning kirjelduse järgi.
- Kasutaja saab lisada, muuta ning kustutada sissetulekuid ja väljaminekuid.
- Kasutaja saab sissetulekute ja väljaminekute juures defineerida järgnevad atribuudid: summa, kategooria, kuupäev ja kirjeldus. Nendest atribuutidest kõik, välja arvatud kirjeldus, on kohustuslikud.
- Kasutajale kuvatakse kogumise eesmärke tema määratud olulisuse järjekorras.
- Kasutaja saab lisada, muuta, märkida tehtuks ning kustutada kogumise eesmärke.
- Kasutaja saab kogumise eesmärkide juures defineerida järgnevad atribuudid: summa, kirjeldus ja olulisus. Kõik atribuudid on kohustuslikud.

Töö vastavust püstitatud funktsionaalsetele nõuetele hinnatakse peatükis „Funktsionaalsetele nõuetele vastavus“.

2.2.2 Mittefunktsionaalsed nõuded

Käesolevas peatükis loetletakse mittefunktsionaalsed nõuded, millele rakendus peab vastama.

Rakendusele seatud mittefunktsionaalsed nõuded on järgmised:

- Rakenduse kasutajaliides on eestikeelne.
- Rakenduse lähtekood on ingliskeelne.

- Rakendus peab olema kasutatav Google Chrome veebibrauseri värskeimas versioonis.
- Kõik kasutaja poolt saadetud sisestatud andmed peavad olema valideeritud.
- Rakendus peab olema kasutaja jaoks arusaadav ja lihtne kasutada.
- Rakenduse kasutajaliides peab olema ekraani suurusega kohalduv.
- Rakendusele peab olema lisatud PWA (*Progressive Web Application*) võimekus.
- Rakendus peab olema ligipääsetav domeeni kaudu.

Töö vastavust püstitatud mittefunktsionaalsetele nõuetele hinnatakse peatükis „Mittefunktsionaalsetele nõuetele vastavus“.

3 Rakenduse realisatsioon

Käesolevas peatükis on kirjeldatud kasutatud tehnoloogiaid ja lahendusi, arenduskeskkonna seadistamist, tööprotsessi, rakenduse arhitektuuri, disaini ja koodi. Rakendust kujundati ja arendati primaarselt mobiilse platvormi tarbeks.

3.1 Kasutatud tehnoloogiad

Käesolevas peatükis antakse lühike ülevaade töös kasutatud tehnoloogiatest ja põhjendatakse nende valikut.

3.1.1 Figma

Kasutajaliidese prototüübi loomiseks kasutati rakendust Figma. See on pilvepõhine disainitööriist, mis on efektiivne ja lihtsustab disainiprotsessi. Figma võimaldab luua nii funktsionaalseid prototüüpe kui ka komponente. Rakendust on võimalik installeerida arvutisse, kuid seda saab kasutada ka otse veebibrauseris. [6]

Figma osutus kasutajaliidese prototüübi tegemisel valituks, sest sel on olemas kõik vajalik, et tõhusalt luua funktsionaalne prototüüp. Samuti saab Figmat kasutada otse brauseris, ilma täiendava tarkvara alla laadimiseta.

3.1.2 IntelliJ IDEA

Projekti arendamisel kasutati IntelliJ IDEA programmi. IntelliJ IDEA arenduskeskkond pakub koodi automaatset lõpetamist, staatilist koodi analüüsi ja refaktoreerimist, võimaldades arendajal olla maksimaalselt efektiivne. IntelliJ IDEAs saab vajadusel alla laadida pistikprogramme, et toetada JavaScripti ja CSS-i kasutamist. [7]

Koodi kirjutamiseks valiti IntelliJ IDEA, autoril on seda programmi mitme aasta jooksul kasutades tekkinud vilumus ning tänu sellele oli võimalik keskenduda rohkem loogika väljamõtlemisele, mitte ideede kirjapanekule.

3.1.3 GitLab

Projekt on salvestatud TalTechi GitLabi keskkonda „iaib“ nimelisse salve. GitLab on avatud *DevOps*’i platvorm, mis võimaldab versioonihaldust hoida paigas ning vajadusel

võtta tagasi kasutusele eelmist versiooni rakendusest. GitLabis on võimalik luua *issue*'sid, mille juurde saab kirja panna projektis arendatava funktsionaalsuse. [8]

Projekti hoidlana kasutati GitLabi, sest selles keskkonnas on võimalik luua *issue*'sid ja *milestone*', on mugav vaadata versiooniajalugu ning saab ideid kirja panna Wiki lehtedele.

3.1.4 Spring Boot

Projekti tagarakendus on loodud, kasutades Java programmeerimiskeelt ja Spring Boot raamistikku. Spring Boot lihtsustab iseseisvate tootmistasemel Springi baasil rakenduste loomist, mida saab lihtsalt käivitada. Enamik Spring Boot rakendusi ei vaja palju konfigureerimist, et projekti tööle panna. [9]

Kuna Spring Boot on kergesti konfigureeritav ning autoril on eelnev kogemus antud raamistiku kasutamisel, osutus see valituks projekti tagarakenduse realiseerimisel.

3.1.5 Vue.js

Projekti eesrakendus on loodud, kasutades programmeerimiskeelt JavaScript ja JavaScripti raamistikku Vue.js. Vue on progressiivne raamistik eesrakenduste loomiseks. Raamistiku tuum on fokusseeritud vaatekihile ja sellele on lihtne lisada teisi raamistikke või olemasolevaid projekte. Vue võimaldab luua keerulisi üheleherakendusi, kui seda kasutatakse koos toetavate raamistike ja modernsete tööriistadega. [10]

Kasutajaliidese kiiremaks arendamiseks on kasutusele võetud BootstrapVue komponentide kogu. See sisaldab üle 85 komponendi, mida on lihtne projektis kasutusele võtta [11].

Lisaks eelnevale on eesrakenduse arendamisel kasutatud ka erinevaid Vue.js pistikprogramme ja stiililehe keelt CSS, et disainida rakendust vastavalt vajadusele.

Vue.js valiti antud projekti eesrakenduse realiseerimiseks, sest ta pakub võimalust mugavalt luua ja kasutada ühefailikomponente, mis muudab koodi loetavamaks ja arusaadavamaks. Vue ühefailikomponent võimaldab ehitada terve komponendi ühes failis, mis koondab nii struktuuri, funktsioone kui ka stiili [12]. Põhiliseks komponentide koguks võeti BootstrapVue, sest selle pakutav disain sobis autori visiooniga rakenduse stiilist.

3.1.6 PostgreSQL

Projekti andmebaasina on kasutusel PostgreSQL. See on avatud lähtekoodiga objekt-relatsiooniline andmebaasisüsteem, mis kasutab ja laiendab SQL keelt. PostgreSQL võimaldab kasutada erinevaid andmetüüpe ja pakub ka erinevaid laiendamisvõimalusi. PostgreSQLil on tugev juurdepääsu kontrollsüsteem, mis muudab selle turvaliseks. PostgreSQL tagab andmete terviklikkuse ning pakub ulatuslikku mahtu. [13]

Valik osutus PostgreSQLi kasuks, võttes arvesse selle võimsust, kasutusvõimalusi ja dokumentatsioonirohkust. Samuti sobib PostgreSQL kasutamiseks nii väiksemate kui ka suuremate ja keerulisemate andmebaaside realiseerimiseks.

3.1.7 Heroku

Nii projekti andmebaas, tagarakendus kui ka eesrakendus on üles seatud Herokus. Heroku on pilveplatvorm, milles on võimalik luua, tarnida, jälgida ja skaleerida aplikatsioone. See võimaldab kiiresti oma rakenduse üles panna. Usaldus ja läbipaistvus on Heroku põhiprintsiibid. [14]

Kuna projekti jaoks kasutatakse Heroku tasuta versiooni, tulenevad sellest mõned piirangud. Andmebaasiga saab luua korraga 20 ühendust ja maksimaalne ridade arv on 10 000.

Projekti taga- ja eesrakendus on Herokus eraldi aplikatsioonidena. Mõlemal on üks *web dyno*. See on sisuliselt serveri eksemplar, mis sisaldab rakenduse lähtekoodi, sõltuvusi ja keskkonnamuutujaid. Kui tasuta versiooni kasutades ei tehta *dyno*'dele ühe tunni jooksul ühtegi päringut, siis siirduvad nad puhkerežiimi. Kui rakendusele teha päring, siis väikese viitega tulevad *dyno*'d automaatselt puhkerežiimist välja. Järgnevad päringud toimuvad viideteta. [15] Rakenduse pika käivitamise ooteaja vältimiseks võeti kasutusele tasulised *Hobby dyno*'d, mille korral ei rakendu puhkerežiim.

Toodangukeskkonnaks valiti Heroku, sest see platvorm pakkus võimalust andmebaas ja projektid kiiresti ning suurema konfigureerimiseta üles panna. Samuti leidus Heroku kasutamise kohta palju informatsiooni, mis lihtsustas keskkonna tundmaõppimist. Samas, Heroku tasuta versiooni probleemideks on esialgne ooteaeg rakenduse käivitumisel ja piiratud andmebaasi ühenduste ning ridade arv.

3.2 Arenduskeskkonna seadistamine

Käesolev peatükk toob välja sammud, mis on vajalikud projekti loomiseks ja selle nii lokaalseks käimapanekuks kui ka tootmiskeskkonnas avaldamiseks.

Arenduskeskkonna seadistamiseks peab olema arvutisse installeeritud järgnev tarkvara:

- Node.js[1] alates versioonist 12
- Versioonikontrollisüsteem Git[2]
- JDK (*Java Development Kit*) 11
- Koodiredaktor, soovitatavalt IntelliJ IDEA[3]

Edasi tuleb projekt oma arvutisse kloonida. Seda on võimalik teha nii käsurealt kui ka läbi IDE (*Integrated Development Environment*).

Käsurealt projekti kloonides tuleb käsku `cd` kasutades liikuda kausta, kuhu soovitakse projekt luua. Seal kaustas olles tuleb sisestada järgnev käsk:

```
git clone https://gitlab.cs.ttu.ee/likutt/iaib.git
```

IntelliJ IDEA-t kasutades tuleb valida menüüst valik *Get from Version Control*. Edasi kuvatakse aken, kus vormi URL (*Uniform Resource Locator*) kohale tuleb sisestada https://gitlab.cs.ttu.ee/likutt/iaib ning seal on ka võimalik ise valida, kuhu kausta projekt kopeeritakse. Kui kõik vajalik on sisestatud, tuleb vajutada nuppu *Clone*.

Samuti on tarvis luua Heroku[4] platvormile kasutajakonto, et teha seal valmis andmebaas ja toodangukeskkond. Mõlema teostamisest tuleb lähemalt juttu järgnevates alapeatükkides.

[1] https://nodejs.org/en/

[2] https://git-scm.com/

[3] https://www.jetbrains.com/idea/

[4] https://signup.heroku.com/login

3.2.1 Andmebaasi seadistamine

Projekt kasutab välist andmebaasi, mis on üles seatud Heroku keskkonnas. Andmebaasi loomiseks tuleb teha järgmised tegevused:

- Logida sisse Heroku platvormile.
- *Dashboard*'il vajutada nuppu *New* ja sealt valida *Create new app.*
- Anda oma rakendusele nimi, valida regioon ning luua rakendus.
- Avada loodud rakenduse *Overview* vaheleht ja sealt valida *Configure Add-ons.*
- Otsingusse sisestada *Heroku Postgres* ja lisada see oma projektile.
- Avada loodud andmebaasis vaheleht *Settings* ja sealt *View Credentials*. Seal olevaid mandaate kasutades saab andmebaasiga luua ühenduse, kasutades näiteks PostgreSQL haldus- ja arendusplatvormi pgAdmin[1].
- Andmebaasi realiseerimiseks on vaja käivitada SQL (*Structured Query Language*) laused, mis on välja toodud Lisas 3.

Eduka andmebaasi loomise korral on see valmis kasutamiseks.

3.2.2 Tagarakenduse arenduskeskkonna seadistamine

Serverirakenduse arenduskeskkonna seadistamiseks tuleb teha järgmised tegevused:

- Kasutades IntelliJ IDEA redaktorit, tuleb installeerida Lombok[2] ja Gradle[3] pistikprogrammid.
- Andmebaasi ühendamiseks tuleb täiendada *application.properties* faili, lisades sinna andmebaasi andmed [16].
- Rakenduses tuleb konfigureerida CORS (*Cross-Origin Resource Sharing*), et lubada rakendusel teha päringuid teistelt domeenidelt [17]. Selle jaoks on vaja defineerida aadressid, millelt on lubatud teha päringuid, ning lubatud meetodid (Joonis 8).

[1] https://www.pgadmin.org/

[2] https://plugins.jetbrains.com/plugin/6317-lombok

[3] https://plugins.jetbrains.com/plugin/13112-gradle

- Käsurealt tuleb `cd` käsuga liikuda *backend* kausta ja seal sisestada käsk `gradlew bootRun`, mis võimaldab rakendust käivitada eelnevalt lähtekoodist tarkvarakomplekti loomata.

```
@SpringBootApplication(exclude = {SecurityAutoConfiguration.class})
public class BackendApplication {

    public static void main(String[] args) {
        SpringApplication.run(BackendApplication.class, args);
    }

    @Bean
    public WebMvcConfigurer corsConfigurer() {
        return new WebMvcConfigurer() {
            @Override
            public void addCorsMappings(CorsRegistry registry) {
                registry.addMapping( pathPattern: "/**")
                        .allowedOrigins("http://localhost:4200",
                                "https://piggybank-e.herokuapp.com",
                                "http://hoiuporsas.com",
                                "https://hoiuporsas.com")
                        .allowedMethods("GET", "PUT", "POST", "DELETE", "OPTIONS");
            }
        };
    }
}
```

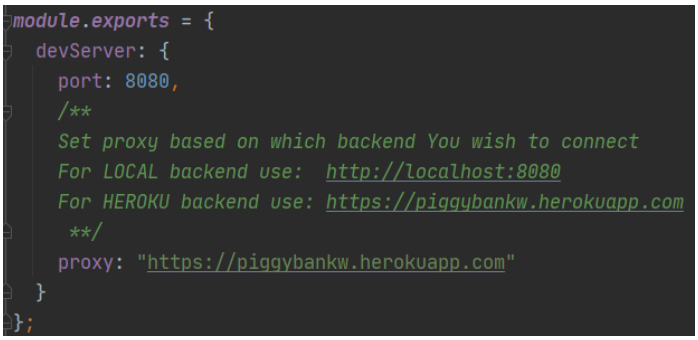
Joonis 8. Tagarakenduses CORS-i konfigureerimine.

Eduka käivitamise korral on server kättesaadav aadressil http://localhost:8080.

3.2.3 Eesrakenduse arenduskeskkonna seadistamine

Kliendipoolse arenduskeskkonna seadistamiseks peab käsurealt liikuma *frontend* kausta. Käesolevas kaustas tuleb esmalt sisestada käsk `npm install`, mis laeb alla paketi koos tema sõltuvustega. Pärast seda tuleb sisestada käsk `npm run serve`, mis palub paketihalduril käivitada failis *package.json* skripti *serve* alla kirjutatud käsu. Käesolevas projektis on *serve* all defineeritud käsk, mis paneb rakenduse tööle pordil 4200.

Lokaalselt projekti eesrakendust käivitades on võimalik muuta kasutatavat tagarakendust. Kasutada saab lokaalselt jooksvat serverit või toodangukeskkonnas paiknevat. Selle muudatuse tegemiseks on vaja *vue.config.js* failis muuta ära *proxy* aadress (Joonis 9).



Joonis 9. Eesrakenduse poolt kasutatava tagarakenduse muutmine.

Eduka käivitamise korral on eesrakendus kättesaadav aadressil http://localhost:4200.

3.2.4 Projekti avaldamine toodangukeskkonnas

Projekti toodangukeskkonnana on kasutusel Heroku. Taga- ja eesrakendused on sinna paigaldatud eraldiseisvate rakendustena.

Terminalist Heroku rakenduste loomise ja haldamise tarbeks võeti kasutusele Heroku CLI (*Command-Line Interface*) [18]. Herokusse projekti juurutamisel kasutati versioonikontrollisüsteemi Git [19].

3.3 Tööprotsess

Tööprotsess sai kohe alustades jagatud viide põhilisse etappi, et projekti arendus jaotada semestri peale ühtlaselt ning paremini aega planeerida. Need etapid olid järgnevad:

- Planeerimine ja Figma prototüübi arendus.
- Staatilisi andmeid kasutava eesrakenduse arendus.
- Andmebaasi ja tagarakenduse loomine.
- Tagarakenduse ühendamine eesrakendusega.
- Viimaste täienduste tegemine ja bakalaureusetöö dokumendi koostamine.

Etappe kirjeldatakse detailsemalt järgnevates alapeatükkides.

Iga etapi jaoks loodi GitLab keskkonnas ka vastav *Milestone*. Nende juures kirjeldati ära *issue*'d, mis tuli ära teha, et saavutada soovitud eesmärk. Koodi *commit*'id tehti, kasutades vastavate *issue*'de numbreid.

Projekti arenduse jooksul konsulteeriti ja tehti kasutajatestimisi kuue algkooliõpilasega, kelle hulgas oli seitsmeaastane, kaheksa-aastane, kaks üheksa- ja kaks üheteistkümneaastast noort. Tegevuste läbiviimiseks oli olemas lapsevanemate nõusolek.

3.3.1 Planeerimine ja Figma prototüüp

Selle kokku peaaegu kolm nädalat kestnud etapiga alustati semestri teisel nädalal. Algselt oli plaanis lõputööna realiseerida laenude haldamisrakendus täiskasvanutele, kuid ilmsiks tuli tõsiasi, et analoogseid rakendusi on palju. Juhendajaga arutluse tulemusena tuli mõte, et fokusseerida rakendus algkooliõpilastele.

Järgnevalt tegi autor intervjuud sihtgruppi kuuluvate noortega ning selgus, et lapsed eriti ei laena asju ja seetõttu puudus vajadus laenude haldamisrakenduse järele. Vestluste käigus tuli ilmsiks asjaolu, et sihtrühma kuuluvad lapsed käivad kooli vahetundide ajal või pärast tundide lõppu ise poes ja ostavad enda raha eest kommi, nätsu, jooke ja nii edasi. Lapsed soovivad samal ajal ka raha koguda, et soetada näiteks mänge, mänguasju või uut mobiiltelefoni. Neljal lapsel puudus täpne ülevaade sellest, kui palju raha neil hetkel kogutud on, kuid nad tahaks seda täpsemalt jälgida. Kahel lapsel oli küll ülevaade oma kogutud rahasummast, kuid nad arvasid samuti, et see võiks paremini jälgitav olla.

Nii kujunes välja antud lõputöö rakenduse idee luua algkooliõpilastele rahaliste vahendite haldamise rakendus. Edasi toimus selle mõtte realiseerimiseks vajalike ettevalmistuste tegemine.

Figma platvormi kasutades koostati rakenduse esialgsed sõrestikmudelid (Lisa 2). Loodud mudeleid valideeriti sihtgrupi esindajatel. Alguses oli nende jaoks arusaamatu, miks prototüübis ei ole võimalik midagi muuta ning kuhu saab vajutada. Pärast sõrestikmudeli eesmärgi ja toimimise selgitamist saadi selle põhimõttest aru. Rakenduse disain oli noorte jaoks loogiline ja nad said rakendusesiseselt navigeerimisega hakkama. Samuti osati välja lugeda informatsiooni nii tehingute kui ka eesmärkide kohta. Algselt oli prototüübis loodud ka statistika leht, kuid kuna see oli nooremate jaoks arusaamatu, jäeti see baasfunktsionaalsusest välja.

Pärast sõrestikmudeli edukat valideerimist uuriti alternatiivseid lahendusi, pandi paika rakenduse nõuded ning töötati välja andmemudel.

3.3.2 Staatiline eesrakendus

Staatilisi andmeid kasutava eesrakenduse arendamine algas semestri kuuendal nädalal ja see kestis kaks nädalat. Otsustati kasutada *frontend first* arendusmetoodikat, sest see võimaldas täpsemalt paika panna, mis andmeid on tagarakendusest vaja kätte saada [20]. Samuti annab see lähenemine võimaluse valideerida valmivat rakendust selle algfaasis ja lihtsustab vajadusel muudatuste tegemist.

Antud etapis sai loodud suur osa eesrakendusest, sealhulgas kasutajakonto loomisleht, sisselogimisleht, pealeht tehingute ja eesmärkide ülevaadetega, tehingute kuvamisleht, tehingu detailvaade, uue tehingu lisamisvaade, tehingute otsimisleht, eesmärkide kuvamisleht, eesmärgi detailvaade, uue eesmärgi lisamisvaade ning kasutajaprofiili leht. Samuti loodi mitmeid komponente, näiteks navigeerimisriba, tehingute ja eesmärkide loendid, nende puudumisteated ning vormid.

Projekti staatiline eesrakendus pandi üles Firebase[1] platvormile ja on kättesaadav aadressilt https://piggybank-bd86a.web.app/. Firebase'i otsustati kasutada, sest see võimaldas staatilise prototüübi kiiresti üles panna ja seda kasutajate peal testida ning valideerida.

Valminud staatiliste andmetega eesrakendus anti igale lapsele katsetamiseks. Samal ajal jälgiti, kuidas nad saavad hakkama rakenduse kasutamisega. Kuuest lapsest viis said iseseisvalt hakkama nii kasutajakonto loomise, sisselogimise kui ka rakendusesiseselt navigeerimisega. Seitsmeaastase jaoks oli kasutajakonto loomine veidi keeruline, aga rakendusesiseselt sai ta kasutamisega hakkama. Üldine tagasiside kasutajaliidesele oli positiivne, lastele meeldis selle välimus ning nad said ka kasutamisega hakkama. Toodi välja, et rakendus võiks enne tehingu ja eesmärgi kustutamist kinnitust küsida, kas ollakse kindel toimingu tegemise osas.

3.3.3 Andmebaasi ja tagarakenduse loomine

Andmebaasi ja tagarakenduse põhifunktsionaalsuse loomisele kulus üks nädal. Andmebaasi seadistamise täpsem kirjeldus on välja toodud peatükis „Arenduskeskkonna seadistamine".

[1] https://firebase.google.com/

Tagarakenduse projektipõhja loomiseks kasutati Spring Initializr[1] tööriista. Selle kasutamisel tuleb täita vorm oma projekti detailide kohta, valida soovitud atribuudid, lisada vajalikud sõltuvused ja laadida alla projektist genereeritud ZIP-fail. Saadud fail pakiti lahti projekti kaustas ja käivitati.

Edasi ühendati Herokus asuv andmebaas tagarakendusega. Kõigepealt realiseeriti tehingute kategooriate päringud, sest nende puhul oli vaja luua ainult pärimisfunktsionaalsus ja kategooriatele ligi pääsemiseks ei olnud vaja kasutajat identifitseerida.

Järgnevalt võeti kasutusele Spring Security, mis on väga kohandatav autentimise ja juurdepääsu kontrollimise raamistik [21]. Spring Security aitab turvata loodud rakendust ning autentida kasutajat. Samaaegselt loodi ka kasutajatega seonduvad klassid ja tarvilikud meetodid. Kasutaja loomisel salvestatakse andmebaasi räsitud parool.

Pärast kasutaja autentimist sai luua tarviliku funktsionaalsuse nii tehingute kui ka kogumiseesmärkide andmebaasist kättesaamiseks, uute objektide lisamiseks, olemasolevate objektide muutmiseks ja kustutamiseks.

Käesoleva etapi raames paigaldati tagarakendus ka Heroku platvormile ja see on kättesaadav aadressilt https://piggybankw.herokuapp.com/.

3.3.4 Eesrakenduse ühendamine tagarakendusega

Etapp, mille põhieesmärgiks oli eesrakenduse ühendamine tagarakendusega, võttes kasutusele dünaamilised andmed andmebaasist, algas semestri kümnendal õppenädalal ja kestis kolm nädalat. Selle etapi jooksul parandati ka ilmnenud vigasid ning lisati tarvilikku funktsionaalsust.

Kõigepealt ühendati kasutaja autentimisega seonduv tegevus tagarakendusega, sest ilma sisse logimata ei ole võimalik rakendust kasutada. Järgnevalt seoti tehingute, kogumiseesmärkide ja tehingukategooriatega seonduvad tegevused serverirakendusega. Selgus, et tagarakendusse jäi lisamata võimekus saada tehinguid ja kogumiseesmärki kätte nende unikaalsete identifikaatorite järgi, ning see võimekus lisati.

[1] https://start.spring.io/

Etapi raames täiendati tehingute, eesmärkide, registreerimis- ja sisselogimisvormide piiranguid vastavalt andmebaasis defineeritud kitsendustele. Lisati ka tulevikku planeeritud tehingute vaade ning realiseeriti kuupäeva vahemiku valimine tehingute kuvamisel. Samuti loodi komponent laadimisspinneri jaoks ja võeti see kasutusele lehtedel, kus võib andmebaasi päringu tegemine aega võtta.

Oodatust keerulisemaks osutus toodangukeskkonnas eesrakenduse ja tagarakenduse ühendamine. Algselt oli plaanis jätkata eesrakenduse ülespaneku kasutamisel Firebase keskkonda, kuid selleks, et sealt teha päringuid välistele API-dele (*Application Programming Interface*), oleks olnud tarvis võtta kasutusele tasuline pakett. Sellel põhjusel otsustati eesrakendus panna üles Heroku keskkonda, nagu projekti tagarakenduski. Raskusi valmistas ka Herokus paikneva taga- ja eesrakenduse CORS-i seadistamine. CORS-i probleemi oleks lahendanud projektide ühte rakendusse panek, kuid neid eraldi rakendustena hoides on hallatavus parem ning see tagab ka tööülesannete loogilise jaotuse.

Selle etapi raames paigaldati eesrakendus, mis teeb päringuid Herokusse paigaldatud tagarakendusele, Heroku platvormile ja see on kättesaadav aadressil https://piggybank-e.herokuapp.com/.

3.3.5 Täienduste teostamine ja dokumendi koostamine

Viimases etapis tehti täiendusi ning parandati rakenduse testimisel ilmnenud probleeme. Samuti koostati bakalaureusetöö dokumenti.

Suuremalt jaolt lisati kasutusmugavust tõstvaid funktsioone, sealhulgas võeti kasutusele kinnitusmodaalid, vormidele lisati tühistamisnupp ning võimekus nupuvajutusega peale lehel alla kerimist tagasi üles liikuda. Tekstialadele rakendati automaatne suuruse muutumine vastavalt kasutaja sisendi pikkusele ning lisati ka kasutada jäänud tähemärkide arvu kuvamine.

Rakendusele sai juurde lisatud PWA võimekus. Progressiivse veebirakenduse saab kasutaja soovil paigaldada oma seadmesse ja kasutada seda brauseri vahekaardi asemel eraldiseisvas aknas [22]. See muudab rakenduse kasutamise mugavamaks, kuid samas jääb alles võimalus pääseda ligi oma andmetele otse brauserist. Kuna PWA ei ole platvormispetsiifiline, saab rakendust paigaldada igale seadmele ja kasutada selleks ühte

koodibaasi [22]. Rakenduse mobiili paigaldamiseks tuleb esmalt see avada veebibrauseris. Google Chrome brauserit kasutades tuleb minna menüüsse ning sealt valida *Install app*.

Parema ligipääsetavuse võimaldamiseks ühendati rakendusega domeen. Domeen ja SSL (*Secure Sockets Layer*) sertifikaat soetati Namecheap[1] platvormilt. Domeeninime kasutamine võimaldab rakendust üles leida, brändi luua ning tekitab ka kasutajates usaldust. SSL on veebilehe kaitsmiseks hädavajalik, sest see tagab privaatsuse, krüpteerides tundlikku informatsiooni, ja andmete terviklikkuse nii veebilehtede kui ka kasutajate jaoks [23]. Rakendus on ligipääsetav domeenilt https://hoiuporsas.com/.

Valminud rakenduse vaated on kuvatud Lisa 4 all.

3.4 Arhitektuur

Arendatav rakendus on ühes projektifailis, mis koosneb kahest projektist. Esimene projekt tegeleb andmebaasist andmepäringute tegemisega. Teine projekt küsib andmeid esimesest projektist ja kuvab neid läbi kasutajaliidese kliendile. Projekt on oma sisult jaotatud taga- ja eesrakenduseks. Projektil on olemas oma andmebaas.

Rakendus on jaotatud kaheks projektiks, vastavalt taga- ja eesrakenduseks, sest see võimaldab neid eraldi hallata. Projektide eraldatus lihtsustab tulevikus projekti edasiarendust. Projekti tagarakendus töötab eraldiseisva REST (*Representational State Transfer*) API-na, mida on võimalik taaskasutada eesrakenduseta.

Alternatiivne võimalus oleks olnud teha rakendus ühe projektina. See oleks lihtsustanud projekti arendust algfaasis ning eesrakenduse suhtlemist tagarakendusega, sest ei oleks olnud vaja kahe eraldiseisva projekti vahel implementeerida CORS-i. Samas, pikemas perspektiivis oleks see lahendus takistanud rakenduse efektiivset edasiarendamist.

3.4.1 Tagarakenduse arhitektuur

Projekti tagarakenduse haldamiseks on kasutatud kihilist arhitektuuri (vt Lisa 5), mis on teostatud vastavalt Spring Boot dokumentatsioonis soovitatavale tüüpilisele paigutusele

[1] https://www.namecheap.com/

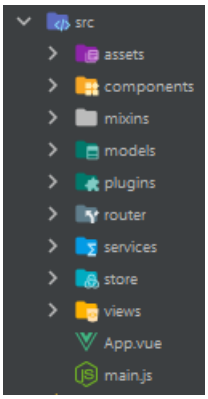
[24]. Selline arhitektuur koondab alamüksused nende tööks vajalike klasside ja meetoditega kompaktselt kokku ning võimaldab nendega individuaalselt tegeleda. Selline eraldatus annab võimaluse arendada igat koodi nii, et tehtud muudatused ei mõjutaks drastiliselt teisi meetodeid ning klasse. Kasutusel olev arhitektuur aitab hoida projekti organiseerituna ja lihtsustab tulevikus edasiarendamist.

Kaustas *src\main\java\ee\taltech\iaib\backend* asub *BackendApplication.java* fail, kus on deklareeritud põhimeetod, mis paneb tagarakenduse tööle. Selles kaustas asuvad ka objektide kaustad, mis sisaldavad vastavate alamüksuste tööks vajalikke klasse ja meetodeid, mis on omakorda tööpõhimõtte järgi kaustadesse jaotatud. Kaustas *src\main\java\ee\taltech\iaib\backend\person* asuvad klassid vastutavad kasutajate registreerimise, sisselogimise ja väljalogimise eest. Kaustas *src\main\java\ee\taltech\iaib\backend\security* asuvate klasside ülesandeks on luua autoriseeritud kasutajatele JWT (*JSON (JavaScript Object Notation) Web Token*), mille alusel kontrollida, et iga kasutaja pääseks ligi ainult iseenda andmetele. Kaustas *src\main\java\ee\taltech\iaib\backend\transaction* asuvad klassid vastutavad tehingutega seonduvate tegevuste eest. Kaustas *src\main\java\ee\taltech\iaib\backend\category* asuvad klassid tegelevad tehingute kategooriatega. Kaustas *src\main\java\ee\taltech\iaib\backend\goal* asuvad klassid teostavad tegevusi kogumiseesmärkidega.

Alamüksuste kaustad sisaldavad omakorda kaustu. Kaustas *controller* olevad klassid haldavad HTTP (*Hypertext Transfer Protocol*) päringuid. Kaustas *model* olevate klasside ülesandeks on esitada andmetabeleid ja nende omavahelisi ühendusi. Kaustas *repository* paiknevad klassid teostavad andmeoperatsioone andmebaasis olevatele andmetele. Kaustas *service* paiknevate klasside ülesanne on teostada äriloogikat ja ühendust andmebaasiga. Kaustas *dto* on klassid, mida kasutatakse andmeedastusobjektidena juhul, kui on vaja andmebaasist saadud andmeid redigeerida või neile informatsiooni lisada. Kaustas *exception* on loodud kohandatud erindid, mida kasutatakse vea ilmnemisel.

3.4.2 Eesrakenduse arhitektuur

Projekti eesrakenduse arhitektuuri (Joonis 10) paikapanemisel on arvestatud sellega, et projekti saaks hõlpsasti edasi arendada ning suuremahuliseks muutumisel oleks see ka lihtsasti hallatav [25].

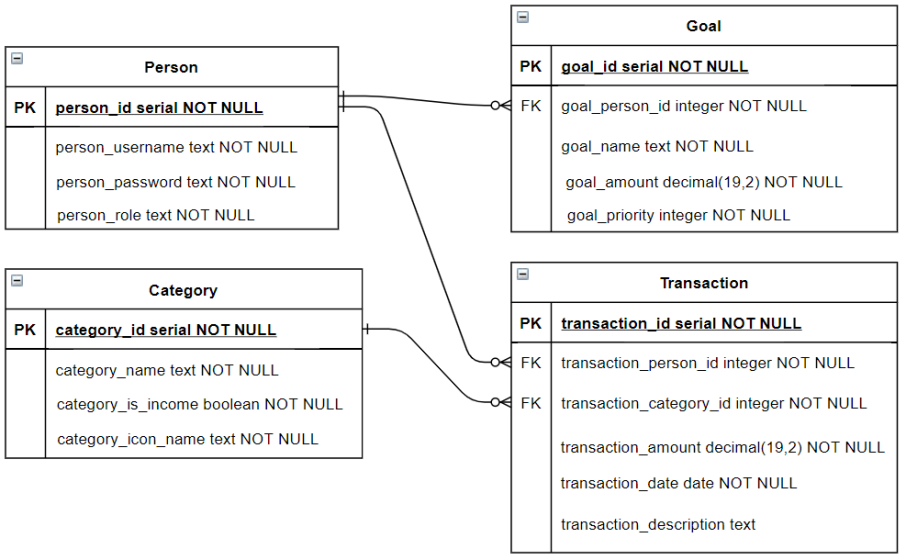


Joonis 10. Projekti eesrakenduse arhitektuur.

Kausta *src\assets* on lisatud rakenduse logo ja sinna saab lisada rakenduses kasutatavaid ressursse. Kaustas *src\components* on Vue korduvkasutavad komponendid, mida kasutatakse vaadetes. Kaustas *src\views* on vaated, mis moodustavad lehtede sisuosa. Kaustas *src\router* kaardistatakse vaated neile vastavatele marsruutidele ja Vue Routeri ülesandeks on kuvada marsruudile vastavat vaadet. Kaustas *src\plugins* registreeritakse kasutusel olevad pistikprogrammid. Kaustas *src\models* on klassid, mis esindavad tagarakendusest tulevaid andmeid. Kaustas *src\services* toimub HTTP päringute teostamiseks Axios instantsi loomine ning kasutaja autoriseerimisega seonduv. Kaustas *src\mixins* on loodud eraldiseisvad klassid eesmärkide, tehingute ja kategooriatega seonduvate päringute tarbeks. Lisaks eelnevale on kõnealuses kaustas ka abimeetodite klass. Kaustas *src\store* on Vuexi haldamisfailid.

3.4.3 Andmebaasi arhitektuur

Andmebaas on jaotatud neljaks tabeliks: *Person*, *Goal*, *Category* ja *Transaction* (Joonis 11). Andmebaasi realiseerimiseks mõeldud SQL laused on toodud välja Lisas 3. Projekti tagarakendus teeb päringuid andmebaasi ja töötleb saadud andmeid vastavalt vajadusele.



Joonis 11. Projekti andmebaasiskeem.

Tabelis *Category* on tehingu kategooriate andmed. Kategooriad on andmebaasi loomisel lisatud ja rakenduse kasutajal neid võimalik muuta ei ole. Tabelis *Person* hoitakse kasutajate andmeid. Kasutajatel on juures atribuut *person_role*, mis võimaldab tulevikus luua juurde erinevaid rolle (näiteks administraator, lapsevanem) lisaks tavakasutajale. Tabelis *Goal* säilitatakse kogumiseesmärkide andmeid. Tabel *Transaction* hoiustab kasutajate sissetulekute ja väljaminekute andmeid.

3.5 Disain

Taga- ja eesrakenduse disainimustrid on erinevad, sest nende realiseerimiseks on kasutatud erinevaid programmeerimiskeeli ja raamistikke. Järgnevad peatükid kirjeldavad mõlemas kasutatavaid mustreid.

3.5.1 Tagarakenduse disain

Projekti tagarakendus kasutab Spring Boot raamistikku. Tagarakendus töötab REST API-na, mis vastab sissetulevatele päringutele.

HTTP meetodite valimisel lähtuti üleüldisest päringumeetodite kasutamisest. GET meetodit kasutatakse andmete pärimiseks. POST meetod on mõeldud andmeüksuse esitamiseks, pärast mida üldiselt kaasneb olekumuudatus või mingi tegevus serveri poolel. Käesolevas projektis lisatakse pärast POST päringut andmebaasi uusi andmeid ning seda kasutatakse ka kasutaja autentimismeetodi ja väljalogimise teostamisel. Meetod PUT muudab andmebaasis olevaid andmeid. DELETE meetod kustutab andmeid. [26]

Tagarakendus tagastab eduka päringu puhul staatuskoodi 200. Kui üritatakse luua kasutajakontot juba kasutusel oleva kasutajanimega, siis tagastatakse staatuskood 400. Kui üritatakse sisse logida väära kasutajanime ja parooli kombinatsiooniga, tagastatakse staatuskood 401. Kui üritatakse pääseda ligi teise kasutaja andmetele, tagastatakse staatuskood 403. [27]

URI (*Uniform Resource Identifier*) teede nimetamisel on lähtutud Microsofti poolt väljatoodud API disaini soovitustest. Nimetamisel on fookus ärilistel üksustel, millele ligipääsu API võimaldab. Samuti tugineti põhimõttele, et päringut ei kirjelda aadress, vaid päringu tegemisel kasutatav HTTP meetod. Eelnevalt nimetatud praktikad teevad aadresside haldamise lihtsamaks, aadressid on intuitiivsemad ning hiljem on API edasiarendamine hõlpsam. [28] Lisa 6 toob välja võimalikud päringud.

3.5.2 Eesrakenduse disain

Projekti eesrakenduse disainimustrid on tingitud Vue.js raamistiku kasutamisest. Järgnevalt tuleb lähemalt juttu mõnest olulisemast kasutatud mustrist.

Üks Vue raamistiku põhifunktsioone on komponentide loomine. Selle jaoks on mitmeid erinevaid võimalusi, millest kaks põhilist on ühefailikomponendid ja malli literaalid ehk *Template Literals*. Malli literaalide kasutamisel puudub süntaksi esiletõstmine ja puudub CSS-i tugi, mis tähendab, et iga komponendi jaoks on vaja luua eraldiseisvad CSS-i failid. See lähenemine on sobilik väikeste ja keskmise suurusega projektide jaoks. [29] Eelnevat arvesse võttes sai kasutusele võetud ühefailikomponendid.

Komponendid peavad ka omavahel suhtlema. Komponentide omavahelise suhtlemise realiseerimiseks on kasutatud *prop*'e ja *event*'e. *Prop*-id antakse vanemkomponendi poolt lapskomponendile. Lapskomponent ei saa *prop*'e muuta, kuid kui vanemkomponent neid muudab, siis rerendertatakse lapskomponendi sisu uuesti. Lapskomponendid saavad oma vahetule vanemkomponendile välja saata *event*'e, et teatada olekumuutusest, andes nii vanemkomponendile võimaluse uuendada lapskomponendi *prop*'e. [30]

Tekib olukordi, kui mõnda komponenti ei ole vaja kuvada. Sellisel juhul on kasutatud komponentide tingimuslikku renderdamist [31].

Eesrakendus peab vastu võtma ja täitma kasutaja antud käsklusi. Selleks on kasutusele võetud *Event Handling* tehnika [32]. Nupuvajutused, vormide muudatused ja teised

toimingud kutsuvad välja meetodi, mis peab vastavalt kasutaja antud käsule täitma oma tööülesande.

3.6 Kood

Projekti koodi kirjutamisel on üritatud järgida *clean code* põhimõtteid. Tähelepanu on pööratud detailidele ja koodi üldisele kvaliteedile.

Ühe väga olulise aspektina on antud muutujatele, funktsioonidele, argumentidele, klassidele, pakettidele, kaustadele ja teistele objektidele sisukad ning tähendusrikkad nimed, mis on tarkvaras kõikjal kasutusel [33, p. 17]. Seda arvesse võttes on oluline, et objektide nimed toetaksid projektis efektiivset navigeerimist. Nimed on valitud selliselt, et need annaksid informatsiooni selle kohta, miks objekt eksisteerib, mida see teeb ning milleks seda saab kasutada, sest heade nimede valik aitab säästa rohkem aega kui nende väljamõtlemisele kulub [33, p. 18]. Kasutatud on otsingut toetavaid pikemaid nimetusi, sest ühe tähe pikkuseid nimesid ja numbrilisi konstante on raske teksti sees üles leida, kuna nad ilmuvad tõenäoliselt igas programmi tekstilõigus [33, p. 22].

Koodi dubleerimist on välditud, sest see suurendab üleüldist koodipagasit, nõuab algoritmi muutmist mitmes erinevas kohas ja võib põhjustada vigasid rakenduse töös [33, p. 48]. Koodi on püütud hoida arusaadavana ning seda on formaaditud, sest see aitab hoida koodi loetavana ja teeb selle hooldamise kergemaks [33, p. 76]. Need on näited mõningatest põhimõtetest, mida on püütud rakenduse arendamisel järgida.

IntelliJ IDEA toetas koodi kvaliteedi parandamist ja selle korrigeerimist. Antud programm pakub selle jaoks mitmeid abifunktsioone, sealhulgas duplikaatide tuvastamist, andmevoo analüüsi, koodi ülevaatust ja kiirparandusi [34]. Lisaks IntelliJ IDEA poolt pakutavale baasfunktsionaalsusele kasutati töös SonarLint pistikprogrammi, mis tuvastab ja aitab lahendada kvaliteediprobleeme [35].

3.6.1 Tagarakenduse kood

Projekti tagarakendus on kirjutatud, kasutades Java programmeerimiskeelt, loodud Spring Boot raamistikku ja see toimib REST API-na, teostades projekti eesrakenduse ja andmebaasi vahelist suhtlust. Lisa 6 toob välja tagarakenduse poolt toetatud päringud.

Rakenduse turvalisuse tagamiseks ja kasutajate autentimiseks on kasutusel Spring Security. API *endpoint*'id on algselt turvatud, kuid rakenduses on *endpoint*'id, mis on vaja teha autoriseerimata kasutajatele kättesaadavaks, sest peab olema võimalik sisse logida ja kasutajakontot luua. Kuna tehingute kategooriad on käesolevas projektis kõikide kasutajate jaoks samad, ei ole tarvis ka neile ligipääsemiseks kasutajat autentida.

Kasutajate autentimine toimub tagarakenduses. Kasutaja sisselogimisel luuakse talle JWT, mis tagastatakse kliendile koos tema kasutajaandmetega. Süsteemis püsib JWT kehtivana 24 tundi ja pärast seda tuleb kasutajal ennast uuesti autentida ehk sisse logida, et tagada turvalisus.

Rakenduse mudelite ja andmeobjektide *boilerplate* koodi vähendamiseks võeti kasutusele Java teek Lombok. Kasutades Lombok teeki, saab vastavaid annotatsioone rakendades automaatselt genereerida näiteks *getter* ja *setter* meetodeid, objektide loomiseks *builder*'it ning palju muud [36].

Tagarakenduse arendusel on oluline konfigureerida CORS ning ühendus andmebaasiga. Seda on kirjeldatud täpsemalt peatükis „Tagarakenduse arenduskeskkonna seadistamine".

Projekti tagarakenduse koodi testimiseks on kirjutatud ühiktestid kategooriate, tehingute, eesmärkide ja inimeste *controller* ning *service* klassidele. Ühiktestimine on tarkvara testimise meetod, millega testitakse tarkvara üksikuid komponente [37]. See aitab tagada koodi vastavust kvaliteedistandarditele ning kiirendada ja lihtsustada vigade avastamist [37].

3.6.2 Eesrakenduse kood

Projekti eesrakendus on loodud JavaScript programmeerimiskeeles, kasutades Vue.js raamistikku. Lisaks on eesrakenduse disainimiseks kasutatud BootstrapVue komponentide kogu, CSS märgistuskeelt ning täiendavaid Vue.js jaoks loodud pistikprogramme. Eesrakendus reageerib brauseri suuruste muutustele, võimaldades rakendust kasutada nii mobiilis, tahvelarvutis, arvutis kui ka suurematel monitoridel.

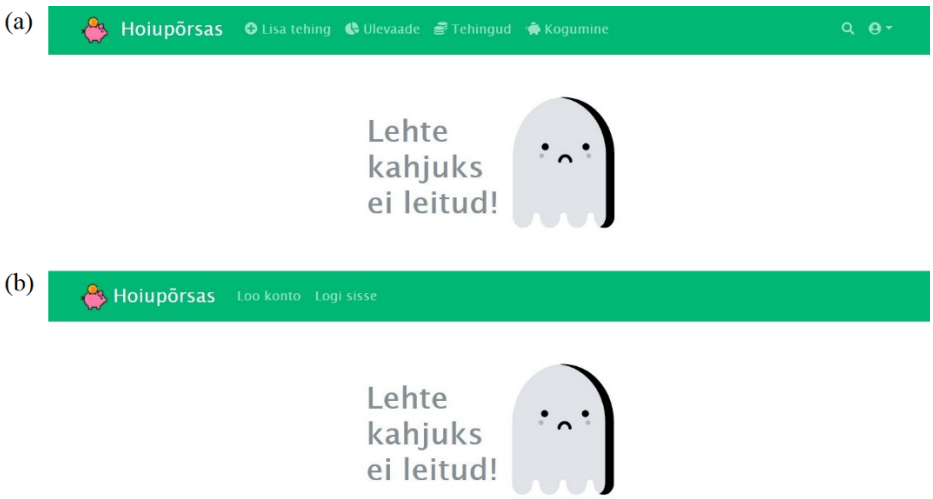
Rakenduses on loodud järgnevad vaated, mille kuvatõmmised on toodud Lisas 4: *Overview*, *PageNotFound*, *LogIn*, *Register*, *Profile*, *TransactionsTimeline*,

FutureTransactions, *SearchTransactions*, *TransactionDetail*, *AddTransaction*, *GoalsOverview*, *GoalDetail* ja *AddGoal*.

Rakenduses on vormide sisendi valideerimiseks kasutusel Vuelidate. See osutus valituks, sest teeb mudelipõhist valideerimist ja võimaldab ilma raske vaevata luua isikupärastatud valideerimismeetodeid [38]. Selleks, et muuta tekstialade kasutamist kasutajate jaoks mugavamaks, võeti kasutusele Vue Textarea Autosize, mis muudab tekstialade suurust automaatselt vastavalt kasutaja sisendi pikkusele [39].

Kasutajaliideses kasutatud ikoonid on pärit Font Awesome tööriistakomplektist, mis lisati projektile, kasutades Vue Fontawesome [40] pistikprogrammi. Kasutamiseks valiti just Font Awesome pakutavad ikoonid, sest need sobisid rakenduse visuaalse visiooniga kokku. Kuna sihtgrupiks on algkooliõpilased, lisati rakendusele isikupärastamiseks ka Vue Kawaii armsate komponentide kogu [41].

Eesrakendus kontrollib igale lehele sisenedes, kas kasutaja on sisse logitud. Juhul, kui kasutaja ei ole sisse logitud, suunatakse ta sisselogimislehele. Ilma sisse logimata pääseb *LogIn*, *Register* ja *PageNotFound* lehtedele, millest viimase menüüriba muutub vastavalt kasutaja olekule (Joonis 12).



Joonis 12. *PageNotFound* kuvatõmmis arvutis: (a) sisseloginud kasutaja, (b) sisselogimata kasutaja

Kasutaja edukal sisselogimisel salvestatakse kliendipoolsesse mälusse [42] päringust tagastatav JWT ja kliendi andmed. Edasiste päringute autentimiseks lisatakse päringu „*Authentication*“ päisesse salvestatud JWT kujul „*Bearer {JWT}*“. Väljalogimisel eemaldatakse JWT ja kliendiandmed mälust.

Andmete pärimine tagarakendusest toimub, kasutades Axios teeki, mis automaatselt muudab saadud päringu JSON objektiks [43]. Eesrakendus kontrollib iga päringu korral, kas see oli autoriseeritud. Juhul, kui päringule tagastatakse 401 HTTP kood, ei ole kasutajal kehtivat JWT-d ja ta suunatakse sisselogimislehele.

4 Valideerimine

Käesolevas peatükis analüüsitakse valminud projekti vastavust seatud nõuetele ning tuuakse välja rakenduse kasutajatestimise tulemused.

4.1 Töö nõuetele vastavus

Peatükis „Nõuded rakendusele“ pandi paika rakenduse peamised nõuded. Nõuetele vastavuse tabel on esitatud Lisas 7. Järgnevalt selgitatakse, mil määral õnnestus nõudeid täita.

4.1.1 Funktsionaalsetele nõuetele vastavus

Üks esmaseid nõudeid oli, et rakendusse oleks võimalik luua kasutajakontot ning logida sisse olemasoleva kasutajana. Antud nõue on täidetud, ilma selleta ei oleks võimalik rakendust kasutada.

Kasutajal on võimalik vaadata oma profiililehte ja näha oma kasutajanime, lisatud tehingute ja eesmärkide arvu. Võiks olla olemas ka lahendus, mis võimaldaks profiili isikupärastada, näiteks lisada profiilipilti ja kustutada oma kasutajakontot. Teostatud on funktsionaalsus, mis võimaldab kasutajal välja logida nii profiililehelt kui ka otse menüüribalt.

Seatud oli nõue, et kasutaja peab nägema ülevaadet oma sissetulekutest ja väljaminekutest. Ülevaate kuvamine on olemas, sealhulgas on tehingud kuvatud kuupäeva järgi kahanevas järjekorras ning listvaates kuvatakse ka iga päeva rahalist kokkuvõtet.

Teostatud on sissetulekute ja väljaminekute kuvamine nädala, kuu, kõik korraga ja kasutaja enda valitud perioodi kaupa. Selguse huvides kuvatakse perioodi rahalist kokkuvõtet, välja arvatud juhul, kui kuvatakse kõiki tehinguid korraga. Kasutajal on võimalik otsida oma tehinguid kategooria nime ja tehingu juurde lisatud kirjelduse järgi. Kasutusmugavuse tõstmiseks oleks võinud otsingu teostamisel rakendada ligikaudset sõnede sobitamise tehnikat, mille korral leitakse üles ka kirjed, mis on sarnased soovitud vastega [44].

Rakenduses on kasutajal võimalik lisada, muuta ja kustutada sissetulekuid ning väljaminekuid. Tehingutel on neli atribuuti: summa, kategooria, kuupäev ja kirjeldus, millest kolm esimest on kohustuslikud.

Kasutaja saab lisada, muuta, märkida tehtuks ning kustutada oma kogumiseesmärke. Kogumiseesmärgil on kolm kohustuslikku atribuuti: summa, kirjeldus ja olulisus. Olulisus on määratud numbriga ning vastavalt sellele järjestatakse kasutaja eesmärgid listvaates olulisuse järgi kahanevalt. Hetkel toimub eesmärgi olulisuse määramine vormis, kasutades vahemiku tüüpi sisendit, kuid see võiks olla lahendatud lohistatava loendi tehnika kasutamisega.

4.1.2 Mittefunktsionaalsetele nõuetele vastavus

Valminud rakenduse kasutajaliides on eestikeelne, sest see on Eestis ligi 900 000 inimese emakeel [45]. Rakenduse lähtekood on ingliskeelne, sest programmeerimismaailmas on inglise keel *lingua franca* ning enamik märksõnu on samuti inglise keeles [46].

Rakendusele on seatud nõue, et see töötaks Google Chrome brauseri värskeimas versioonis. Veebilehe Statcounter andmete kohaselt on Google Chrome kõige populaarsem veebibrauser viimase kaheteistkümne kuu jooksul [47]. Töö kirjutamise hetkel on brauseri viimane väljaantud versioon 90.0.4430.85 ja rakendus töötab seda kasutades edukalt.

Rakenduse kasutuskogemuse seisukohalt on oluline, et kasutaja sisestatud andmed oleks valideeritud. Andmed valideeritakse juba eesrakenduses enne tagarakendusele saatmist ja ebakõlade korral kuvatakse kasutajale vastavaid veateateid. Samuti on lisatud kontrollid ka tagarakendusse ning andmebaasis on loodud vajalikud kitsendused.

Rakendus on primaarselt mõeldud mobiilis kasutamiseks, kuid peab olema kasutatav ka arvutis. Seda arvesse võttes on rakendus loodud nii, et kasutajaliides oleks ekraani suurusega kohalduv. Rakendus on ligipääsetav domeenilt ning sellele on lisatud ka PWA võimekus.

Nõue, et rakendus peab olema kasutaja jaoks arusaadav ja lihtne kasutada, sai täidetud. Järgnevas peatükis on täpsemalt kirjeldatud kasutajatestimise tulemusi, mille põhjal ka hinnati nõude täidetuks osutumist.

4.2 Rakenduse kasutajatestimine

Töö kirjutamise hetkel kehtinud eriolukorra tõttu ei olnud võimalik esialgselt planeeritud testimist teha. Algselt oli plaanis anda rakendus algkooliõpilastele nädalaks ajaks kasutada, et nad saaksid jooksvalt märkida üles oma sissetulekuid, väljaminekuid ning seada omale kogumiseesmärke, kuid distantsõppe tõttu ei kulutanud nad raha nii, nagu enne kaugõppe kehtestamist.

Tõenäosus, et kasutajatestimise ajal esineb viga, on 31%. Testides rakendust viie kasutajaga ilmneb 85% kasutajaliidese probleemidest. Poissoni jaotus 31% protsendilise binoomtõenäosusega illustreerib, et lisades rohkem kui viis kasutajat testimisrühma, väheneb tulem drastiliselt – mida rohkem kasutajaid lisada testimisgruppi, seda vähem informatsiooni saadakse. [48]

Eelnevat arvesse võttes tehti rakendus kättesaadavaks kuuele noorele ja jälgiti, kuidas nad selle kasutamisega toime tulevad. Testijateks olid seitsmeaastane, kaheksa-aastane, kaks üheksa- ja kaks üheteistkümneaastast noort. Kasutajatestimine viidi läbi nutitelefonides, kasutades toodangukeskkonda üles pandud rakendust läbi Google Chrome veebibrauseri. Testimisel paluti noortel meenutada oma viimaseid sissetulekuid ja väljaminekuid või panna kirja hüpoteetilised tehingud. Samuti esitati vajadusel suunavaid küsimusi ja hüpoteetilisi olukorra kirjeldusi.

Esimeseks sammuks rakenduse kasutamisel on kasutajakonto loomine ja pärast seda sisselogimine. Kõik testijad said kenasti lehele registreeritud ja vastloodud kasutajakontoga sisse logitud. Üks üheksa-aastane proovis luua samanimelist kasutajakontot uuesti ja oli rahul, et seda ei saa teha.

Pärast sisselogimist kuvatakse ülevaateleht, kuhu algselt ei ole ühtki tehingut ega eesmärki lisatud. Neli last alustas rakenduse kasutamisel tehingute lisamisest, üks alustas eesmärgi lisamisest ja üks hakkas menüüd läbi vaatama ning käis olemasolevad lehed läbi.

Täpsemad testitud kasutusjuhud ja nende tulemused on välja toodud Lisas 8 esitatud tabelis. Toetudes nendele tulemustele saab väita, et rakendus on algkooliõpilastele arusaadav ja lihtne kasutada. Lapsed said iseseisvalt hakkama rakenduses navigeerimisega ja erinevate toimingute teostamisega. Suunavaid küsimusi ja

hüpoteetilisi olukordi esitati neile, kes ise ei läbinud antud kasutusjuhtu. See oli vajalik, et kontrollida, kas kõik testijad leiavad kõnealuse funktsionaalsuse üles ja oskavad seda kasutada.

Laste arvates nägi rakendus hea välja. Eriti meeldisid neile Vue Kawaii komponendid. Toodi välja, et võiks olla võimalik muuta rakenduse värviteemasid ning lisada profiilipilti. Kuuest lapsest neli sooviksid rakendust ka tulevikus edasi kasutada, ülejäänud kaks tõid välja, et nad ei viitsiks ikkagi kõike kirja panema hakata.

5 Võimalikud edasiarendused

Bakalaureusetöö raames valmis baasfunktsionaalsusega rakendus, mis on paigaldatud veebilehele https://hoiuporsas.com/. Realiseeritud on järgnevad funktsionaalsused: kasutajakonto loomine, sisselogimine ning tehingute ja kogumiseesmärkidega seonduvad tegevused. Samuti on rakendusele lisatud PWA võimekus. Põhiline äriloogika on olemas ja rakendust on praktikas juba võimalik kasutada, kuid kasutajakogemuse parandamiseks oleks soovitatav rakendusele teha mitmeid edasiarendusi.

Primaarsed lisad, mida võiks tulevikus arendada, on järgnevad:

- Kasutajaprofiili isikupärastamine.
- Erinevate värviteemade valikuvõimalus.
- Lapsevanema rolli loomine.
- Kasutajale statistika kuvamine.
- Konto loomine sotsiaalmeedia kasutajakontoga.
- Arhiiv täidetud eesmärkidele.
- Võimaldada kasutajal tehingute kategooriaid konfigureerida ja isikupärastada.
- Eesmärkide järjestamine lohistatava loendi põhimõttel.
- Kasutajakonto kustutamine.
- Rakenduse lokaliseerimine
- Koodi automaatne avaldamine toodangukeskkonnas.

Lisaks väljatoodud funktsionaalsusele on kindlasti vaja tegeleda jooksvalt ilmnevate probleemidega.

6 Kokkuvõte

Käesoleva töö eesmärgiks oli realiseerida rakendus, mis võimaldab rahalisi vahendeid hallata ja kogumiseesmärke registreerida. Loodud rakendus on eelkõige mõeldud algkooliõpilastele, kelle jaoks olemasolevad lahendused on kasutamiseks liiga keerulised.

Töö käigus analüüsiti olemasolevaid rakendusi ja suheldi rakenduse potentsiaalsete kasutajatega. Analüüsi ja kasutajatega vestluse põhjal prototüübiti ja loodi *full-stack* rahaliste vahendite haldamise rakendus. Rakenduse prototüüpimiseks kasutati Figmat ning implementeerimiseks Vue.jsi, Spring Booti ja PostgreSQLi.

Töö on jaotatud kaheks väiksemaks projektiks. Projekti tagarakendus tegeleb loodud andmebaasist andmepäringute tegemisega. Projekti eesrakendus küsib andmeid tagarakendusest ja kuvab neid läbi kasutajaliidese kliendile. Arenduse käigus on lähtutud kirjanduses ning internetis leiduvatest arhitektuuri, disaini ja koodi kirjutamise reeglitest ja headest tavadest.

Lõputöö käigus valmis baasfunktsionaalsust omav rakendus. Loodud rakendus erineb konkurentidest, sest on orienteeritud algkooliõpilastele ja seetõttu on see tehtud võimalikult arusaadavaks. Samuti on rakenduse kasutajaliides eestikeelne.

Valminud rakendus on ligipääsetav aadressil https://hoiuporsas.com/ ja lähtekoodi repositoorium on saadaval aadressil https://gitlab.cs.ttu.ee/likutt/iaib.

Kasutatud kirjandus

[1] Rahakool, “Rahakool koduleht,” TJ Süsteemiarendus OÜ, [Online]. Available: https://rahakool.ee/rk/index.php. [Accessed 21 4 2021].

[2] Perekonna Eelarve, “Perekonna Eelarve koduleht,” [Online]. Available: https://www.pere-eelarve.eu/. [Accessed 21 4 2021].

[3] Spendee, “Spendee koduleht,” [Online]. Available: https://www.spendee.com/. [Accessed 21 4 2021].

[4] Spendee, “Terms of Use,” [Online]. Available: https://files.spendee.com/terms-and-conditions/Terms%20of%20Use.pdf. [Accessed 21 4 2021].

[5] J. Nielsen, “10 Usability Heuristics for User Interface Design,” 15 11 2020. [Online]. Available: https://www.nngroup.com/articles/ten-usability-heuristics/. [Accessed 21 4 2021].

[6] B. Kopf, “The Power of Figma as a Design Tool,” [Online]. Available: https://www.toptal.com/designers/ui/figma-design-tool. [Accessed 21 4 2021].

[7] JetBrains s.r.o., “IntelliJ IDEA overview,” [Online]. Available: https://www.jetbrains.com/help/idea/discover-intellij-idea.html. [Accessed 21 4 2021].

[8] GitLab, “GitLab koduleht,” [Online]. Available: https://about.gitlab.com/ . [Accessed 21 4 2021].

[9] Spring, “Spring Boot,” [Online]. Available: https://spring.io/projects/spring-boot. [Accessed 21 4 2021].

[10] Vue.js, “Introduction,” [Online]. Available: https://vuejs.org/v2/guide. [Accessed 21 4 2021].

[11] BootstrapVue, “BootstrapVue koduleht,” [Online]. Available: https://bootstrap-vue.org/. [Accessed 21 4 2021].

[12] B. Wozniewicz, “freeCodeCamp,” [Online]. Available: https://www.freecodecamp.org/news/how-to-create-a-vue-js-app-using-single-file-components-without-the-cli-7e73e5b8244f/. [Accessed 11 5 2021].

[13] PostgreSQL, “About,” [Online]. Available: https://www.postgresql.org/about/ . [Accessed 22 4 2021].

[14] Heroku, “What is Heroku?,” [Online]. Available: https://www.heroku.com/what. [Accessed 21 4 2021].

[15] M. Pundsack, “App Sleeping on Heroku,” 20 6 2013. [Online]. Available: https://blog.heroku.com/app_sleeping_on_heroku. [Accessed 21 4 2021].

[16] MBcoder, “Connecting to Heroku Postgres from Spring Boot,” 25 6 2019. [Online]. Available: https://mbcoder.com/connecting-to-heroku-postgres-from-spring-boot/. [Accessed 23 4 2021].

[17] MDN contributors, “Cross-Origin Resource Sharing (CORS),” [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS. [Accessed 24 4 2021].

[18] Heroku, “The Heroku CLI,” [Online]. Available: https://devcenter.heroku.com/articles/heroku-cli. [Accessed 23 4 2021].

[19] Heroku, “Deploying with Git,” [Online]. Available: https://devcenter.heroku.com/articles/git. [Accessed 23 4 2021].

[20] “Frontend first development,” [Online]. Available: https://www.frontendfirstdevelopment.com/. [Accessed 23 4 2021].

[21] Spring, “Spring Security,” [Online]. Available: https://spring.io/projects/spring-security. [Accessed 23 4 2021].

[22] P. L. S. Richard, “What are Progressive Web Apps?,” [Online]. Available: https://web.dev/what-are-pwas/. [Accessed 7 5 2021].

[23] S. Shopper, “Why SSL? The Purpose of using SSL Certificates,” [Online]. Available: https://www.sslshopper.com/why-ssl-the-purpose-of-using-ssl-certificates.html. [Accessed 7 5 2021].

[24] Spring, “Locating the Main Application Class,” [Online]. Available: https://docs.spring.io/spring-boot/docs/current/reference/html/using-spring-boot.html#using-boot-locating-the-main-class. [Accessed 22 4 2021].

[25] M. Minasyan, “Vue.js architecture for large scale projects,” [Online]. Available: https://medium.com/js-dojo/vue-js-architecture-for-large-scale-projects-7942dd3ce9f8. [Accessed 22 4 2021].

[26] MDN contributors, “HTTP request methods,” [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods. [Accessed 22 4 2021].

[27] “HTTP Status Codes,” [Online]. Available: https://restfulapi.net/http-status-codes/. [Accessed 22 4 2021].

[28] Microsoft, “Web API design,” [Online]. Available: https://docs.microsoft.com/en-us/azure/architecture/best-practices/api-design. [Accessed 22 4 2021].

[29] Vue.js, “Single File Components,” [Online]. Available: https://vuejs.org/v2/guide/single-file-components.html. [Accessed 22 4 2021].

[30] Vue Patterns, “Component Communication,” [Online]. Available: https://learn-vuejs.github.io/vue-patterns/patterns/#component-communication. [Accessed 22 4 2021].

[31] Vue Patters, “Component Conditional Rendering,” [Online]. Available: https://learn-vuejs.github.io/vue-patterns/patterns/#component-conditional-rendering. [Accessed 22 4 2021].

[32] Vue.js, “Event Handling,” [Online]. Available: https://vuejs.org/v2/guide/events.html. [Accessed 22 4 2021].

[33] R. C. Martin, Clean Code: A Handbook of Agile Software Craftsmanship, Prentice Hall, 2009.

[34] JetBrains s.r.o., “IntelliJ IDEA Features,” [Online]. Available: https://www.jetbrains.com/idea/features/. [Accessed 24 4 2021].

[35] SonarLint, “SonarLint koduleht,” [Online]. Available: https://www.sonarlint.org/. [Accessed 24 4 2021].

[36] Project Lombok, “Lombok features,” [Online]. Available: https://projectlombok.org/features/all. [Accessed 24 4 2021].

[37] Forte Group, “The importance of unit testing, or how bugs found in time will save you money,” [Online]. Available: https://fortegrp.com/the-importance-of-unit-testing/. [Accessed 16 5 2021].

[38] Vuelidate, “Vuelidate koduleht,” [Online]. Available: https://vuelidate.js.org/. [Accessed 24 4 2021].

[39] devstark-com, “vue-textarea-autosize koduleht,” [Online]. Available: https://github.com/devstark-com/vue-textarea-autosize. [Accessed 24 4 2021].

[40] andhovesyan, “vue-fontawesome koduleht,” [Online]. Available: https://github.com/andhovesyan/vue-fontawesome. [Accessed 24 4 2021].

[41] youngtailors, “vue-kawaii koduleht,” [Online]. Available: https://github.com/youngtailors/vue-kawaii. [Accessed 24 4 2021].

[42] Vue.js, “Client-Side Storage,” [Online]. Available: https://vuejs.org/v2/cookbook/client-side-storage.html. [Accessed 24 4 2021].

[43] axios, “axios koduleht,” [Online]. Available: https://github.com/axios/axios. [Accessed 24 4 2021].

[44] Z. A. Khan, “Algorithms for Approximate String Matching,” [Online]. Available: https://www.linkedin.com/pulse/algorithms-approximate-string-matching-zafar-ali-khan-1/. [Accessed 26 4 2021].

[45] A. T. K. Raid, “Eestis on kasvanud emakeelte mitmekesisus,” [Online]. Available: https://www.stat.ee/et/uudised/2019/03/14/eestis-on-kasvanud-emakeelte-mitmekesisus. [Accessed 23 3 2021].

[46] Y Studios, “The Language of Codes : Why English is the Lingua Franca of Programming,” [Online]. Available: https://ystudios.com/insights-passion/codelanguage. [Accessed 26 4 2021].

[47] Statcounter, “Browser Market Share Worldwide,” [Online]. Available: https://gs.statcounter.com/browser-market-share. [Accessed 26 4 2021].

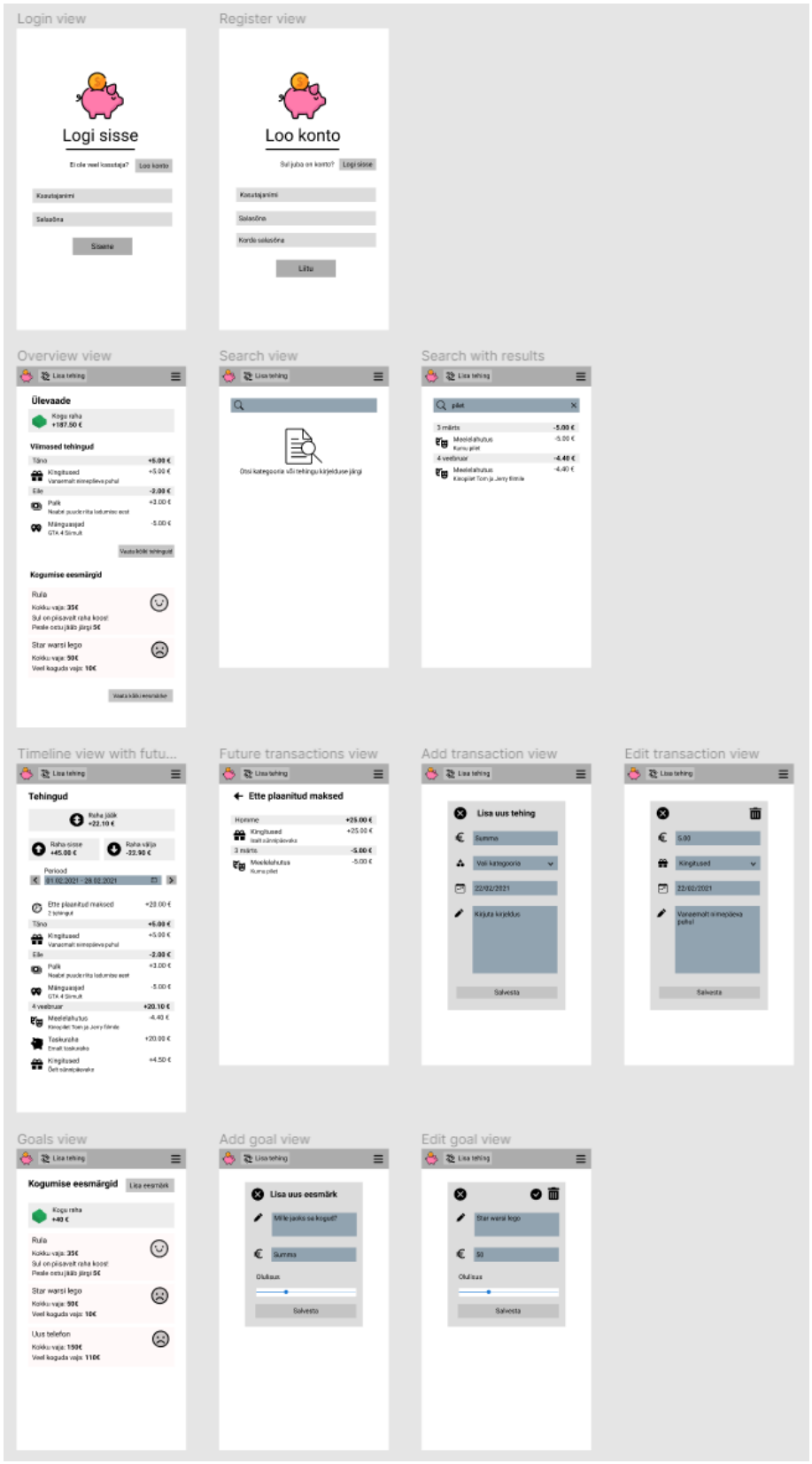
[48] E. Martin, “InVision,” [Online]. Available: https://www.invisionapp.com/inside-design/ux-usability-research-testing/. [Accessed 11 5 2021].

Lisa 1 – Alternatiivsete lahenduste analüüs kasutades Jakob Nielseni 10 kasutatavuse heuristikat

Kasutatavuse heuristika	**Rahakool**	**Perekonna Eelarve**	**Spendee**
Ülevaade süsteemi staatusest	Puudulik	Hea	Väga hea
Seos rakenduse ja reaalse maailma vahel	Väga hea	Väga hea	Väga hea
Kasutajapoolne kontroll ja vabadus	Täitmata	Rahuldav	Hea
Järjepidevus ja standardid	Väga hea	Väga hea	Väga hea
Vigade ennetamine	Puudulik	Puudulik	Hea
Pigem äratundmine kui meenutamine	Rahuldav	Väga hea	Väga hea
Paindlikkus ja kasutusefektiivsus	Rahuldav	Hea	Väga hea
Esteetiline ja minimalistlik kujundus	Täitmata	Väga hea	Väga hea
Aita kasutajatel ära tunda, diagnoosida ja taastuda vigadest	Rahuldav	Rahuldav	Rahuldav
Abiinfo ja dokumentatsioon	Hea	Täitmata	Väga hea

Lisa 2 – Rakenduse esialgsed sõrestikmudelid

Rakenduse sõrestikmudel Figmas.



Lisa 3 – Andmebaasi realiseerimislaused PostgreSQL-is

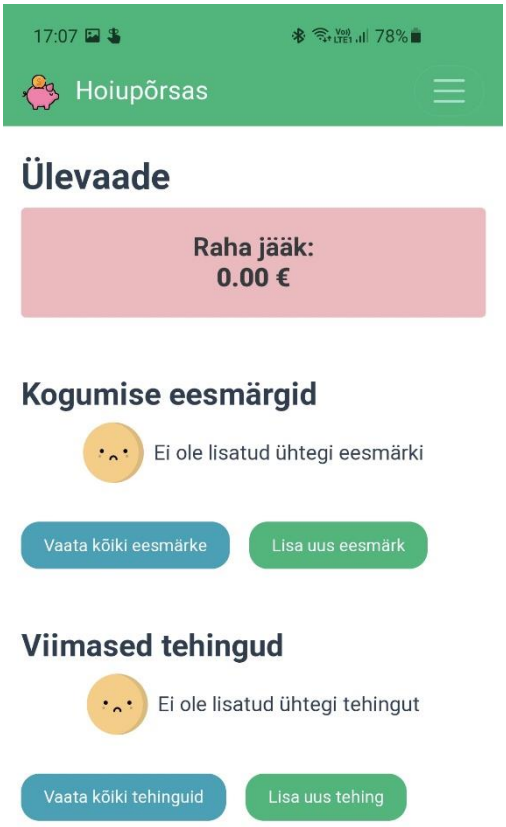
```
SET client_encoding=UTF8;

CREATE TABLE Category (
category_id serial NOT NULL,
category_name text NOT NULL,
category_is_income boolean NOT NULL,
category_icon_name text NOT NULL,
CONSTRAINT PK_Category PRIMARY KEY (category_id),
CONSTRAINT CHK_Category_category_name_is_up_to_50_marks CHECK
(LENGTH(category_name) <=50),
CONSTRAINT CHK_Category_category_name_is_not_empty_text CHECK (category_name
!~ '^[[:space:]]*$'));

INSERT INTO Category (category_name, category_is_income, category_icon_name)
VALUES
('Söök ja jook', false, 'utensils'),
('Kool', false, 'school'),
('Tervis', false, 'heartbeat'),
('Isiklik', false, 'user'),
('Šoppamine', false, 'shopping-cart'),
('Ehted', false, 'gem'),
('Riided', false, 'tshirt'),
('Jalanõud', false, 'shoe-prints'),
('Elektroonika', false, 'mobile-alt'),
('Meelelahutus', false, 'theater-masks'),
('Raamatud', false, 'book'),
('Mängud', false, 'gamepad'),
('Mänguasjad', false, 'dice'),
('Hobid', false, 'star'),
('Kino', false, 'video'),
('Transport', false, 'car-alt'),
('Heategevus', false, 'hand-holding-heart'),
('Kingitused', false, 'gift'),
('Muu', false, 'minus-circle'),
('Taskuraha', true, 'coins'),
('Kingitused', true, 'gift'),
('Palk', true, 'wallet'),
('Auhind', true, 'trophy'),
('Muu', true, 'plus-circle');

CREATE TABLE Person (
person_id serial NOT NULL,
person_username text NOT NULL,
person_password text NOT NULL,
person_role text NOT NULL,
CONSTRAINT PK_Person PRIMARY KEY (person_id),
CONSTRAINT AK_Person_username_is_unique UNIQUE (person_username),
```

```
CONSTRAINT CHK_Person_username_is_up_to_50_marks CHECK
(LENGTH(person_username) <= 50),
CONSTRAINT CHK_Person_username_is_not_empty_text CHECK (person_username !~
'^[[:space:]]*$'));
CREATE UNIQUE INDEX IX_Person_Username_is_unique ON Person
(Upper(person_username));

CREATE TABLE Goal (
goal_id serial NOT NULL,
goal_person_id integer NOT NULL,
goal_name text NOT NULL,
goal_amount decimal(19,2) NOT NULL,
goal_priority integer NOT NULL,
CONSTRAINT PK_Goal PRIMARY KEY (goal_id),
CONSTRAINT CHK_Goal_goal_name_is_up_to_50_marks CHECK (LENGTH(goal_name) <=
50),
CONSTRAINT CHK_Goal_goal_name_is_not_empty_text CHECK (goal_name !~
'^[[:space:]]*$'),
CONSTRAINT CHK_Goal_goal_amount_bigger_or_equal_to_0 CHECK (goal_amount >=
0),
CONSTRAINT FK_Goal_goal_person_id FOREIGN KEY (goal_person_id) REFERENCES
Person (person_id)
       ON DELETE No Action ON UPDATE Cascade);

CREATE TABLE Transaction (
transaction_id serial NOT NULL,
transaction_person_id integer NOT NULL,
transaction_category_id integer NOT NULL,
transaction_amount decimal(19,2) NOT NULL,
transaction_date date NOT NULL,
transaction_description text,
CONSTRAINT PK_Transaction PRIMARY KEY (transaction_id),
CONSTRAINT CHK_Transaction_description_is_up_to_280_marks
       CHECK (LENGTH(transaction_description) <= 280),
CONSTRAINT FK_Transaction_transaction_person_id FOREIGN KEY
(transaction_person_id)
       REFERENCES Person (person_id) ON DELETE No Action ON UPDATE Cascade,
CONSTRAINT FK_Transaction_transaction_category_id FOREIGN KEY
(transaction_category_id)
       REFERENCES Category (category_id) ON DELETE No Action ON UPDATE
Cascade);
```

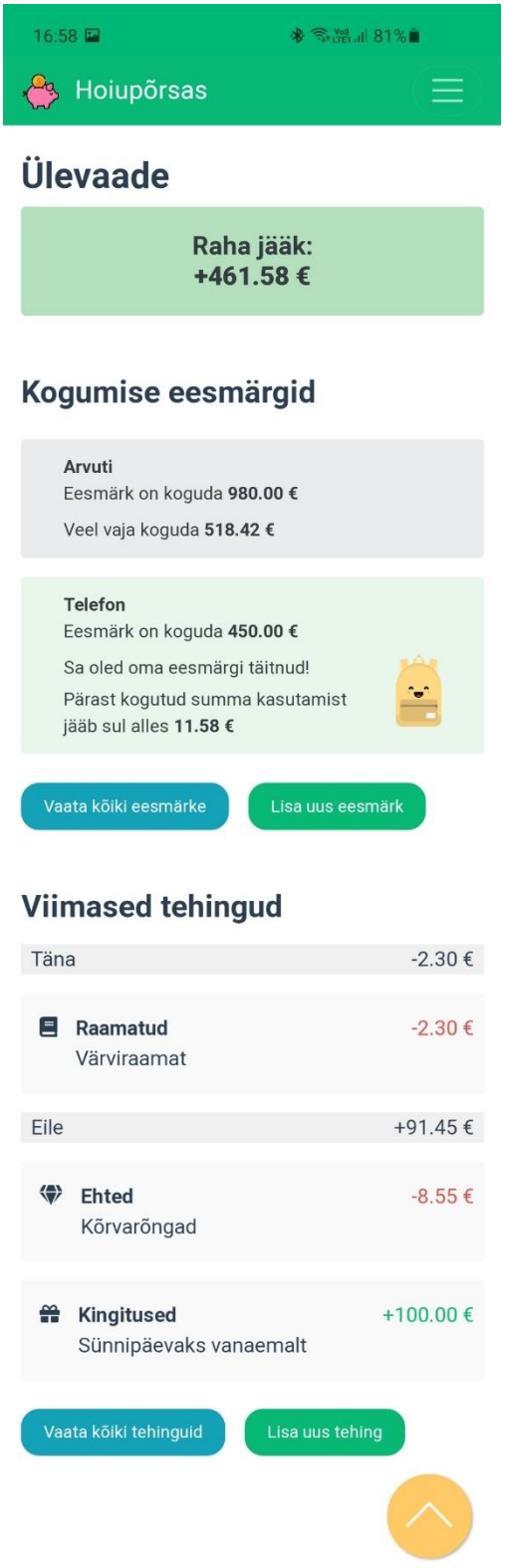
Lisa 4 – Rakenduse kuvatõmmised

Sisselogimisvaade mobiilis – *LogIn*

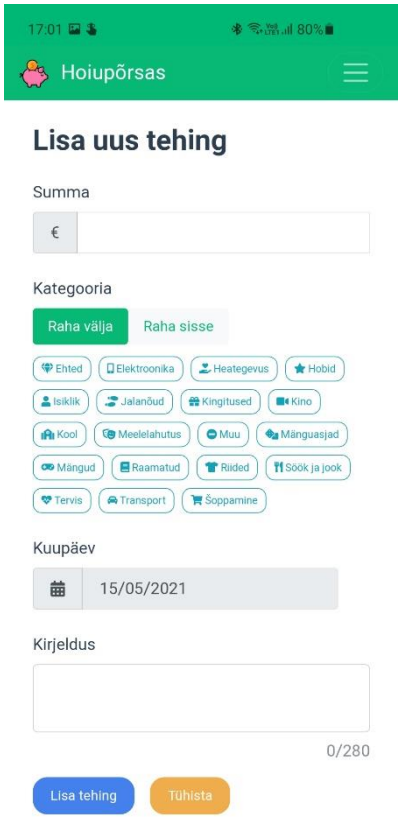
Registreerimisvaade mobiilis – *Register*



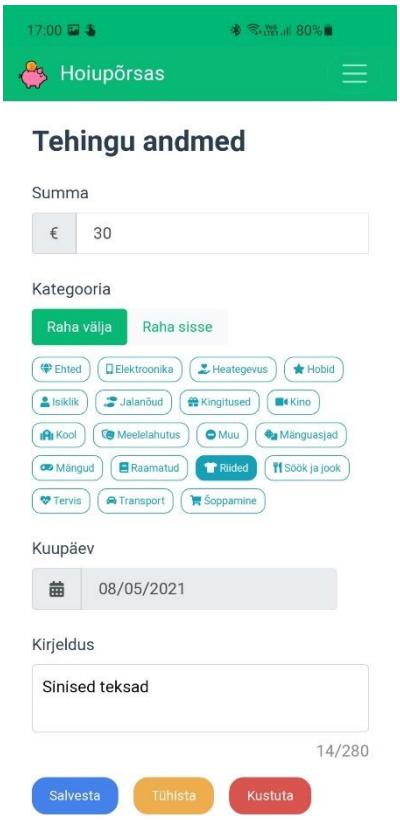
Tehingute ja eesmärkideta ülevaateleht mobiilis – *Overview*



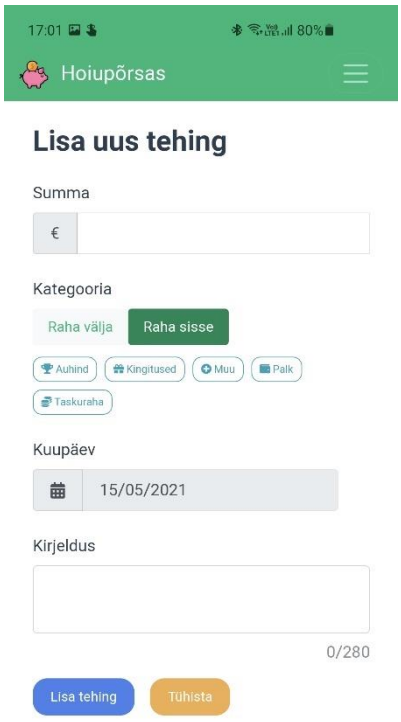
Tehingute ja eesmärkidega ülevaateleht mobiilis – *Overview*



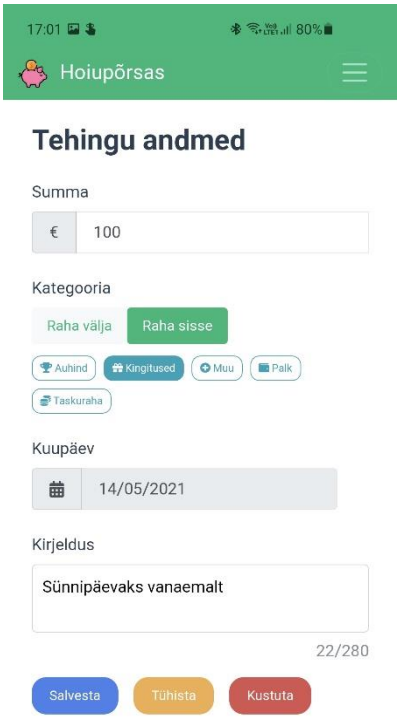
Kulutuse lisamisvaade mobiilis – *AddTransaction*



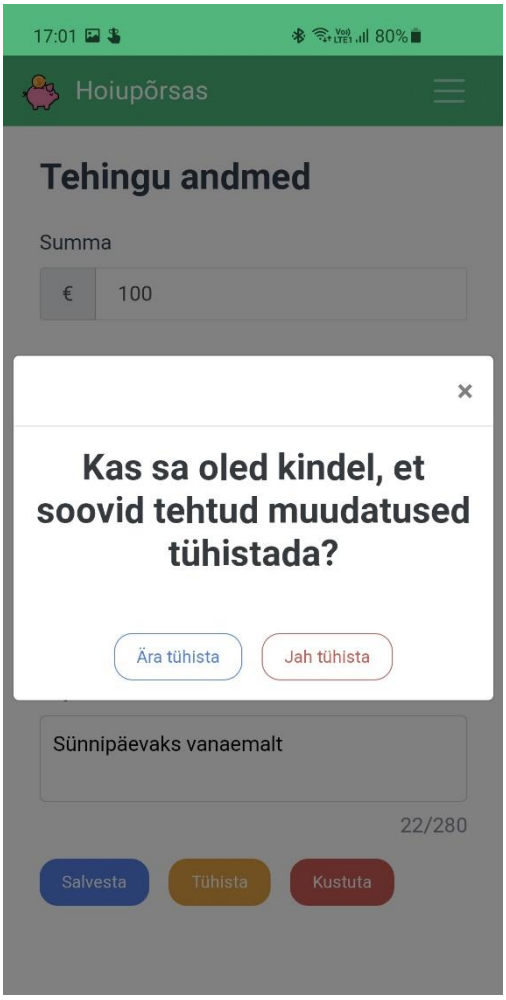
Kulutuse detailvaade mobiilis – *TransactionDetail*



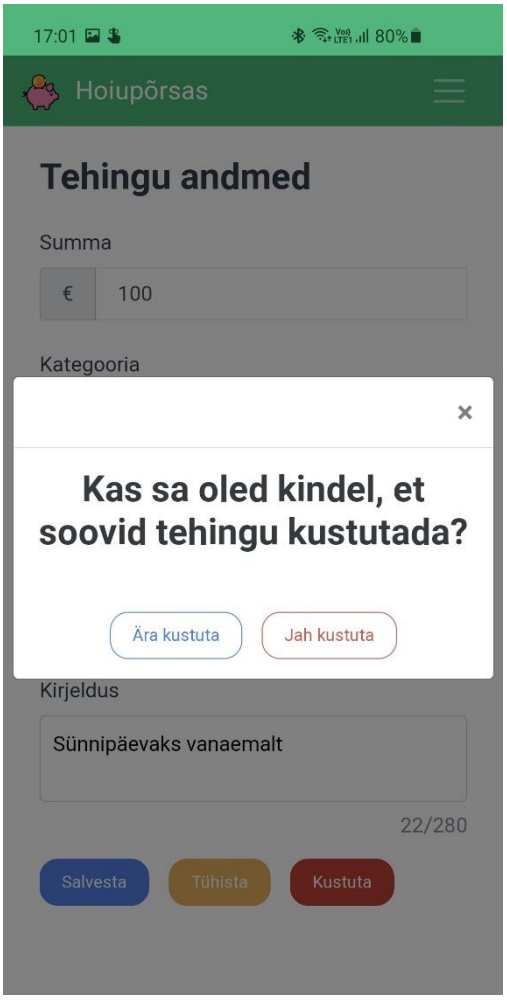
Sissetuleku lisamisvaade mobiilis – *AddTransaction*



Sissetuleku detailvaade mobiilis – *TransactionDetail*

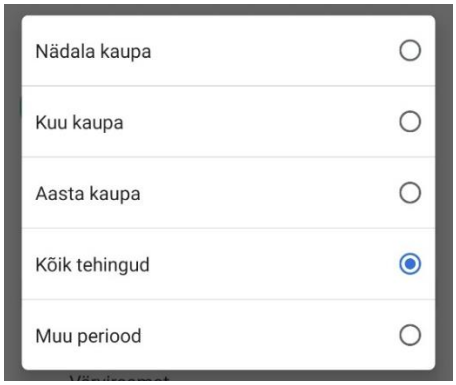


Tehingu muudatuste tühistamise kinnitusmodaal mobiilis – *TransactionDetail*



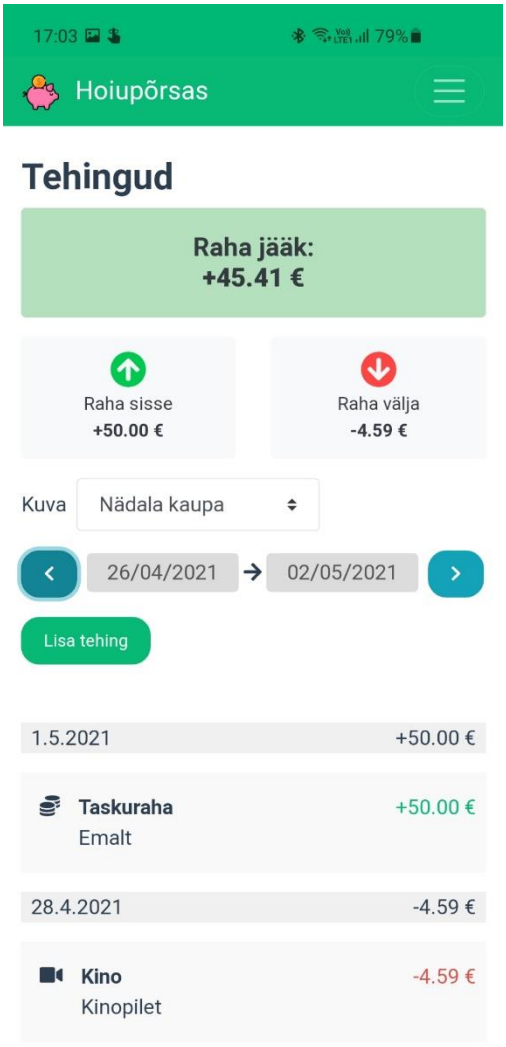
Tehingu kustutamise kinnitusmodaal mobiilis – *TransactionDetail*

Tühi tehinguteleht mobiilis – *TransactionsTimeline*

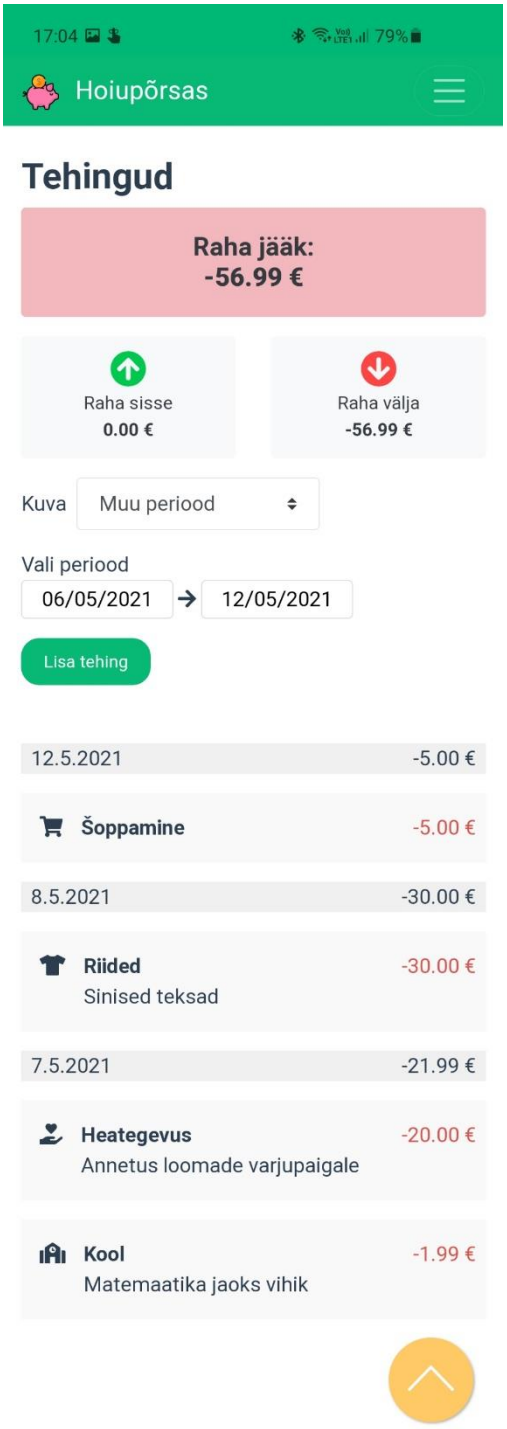


Tehingute perioodi valik mobiilis – *TransactionsTimeline*

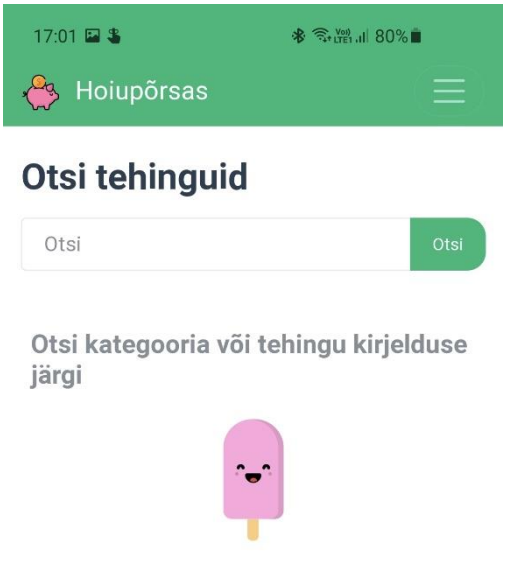
Tehingutega tehinguteleht mobiilis – *TransactionsTimeline*



Nädala tehinguteleht mobiilis – *TransactionsTimeline*



Kasutaja määratletud perioodi tehinguteleht mobiilis – *TransactionsTimeline*

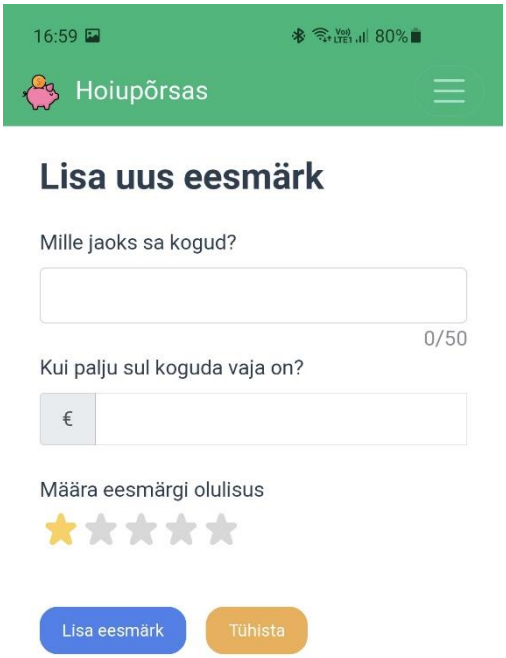


Tehingute otsingulehe avavaade mobiilis – *SearchTransactions*

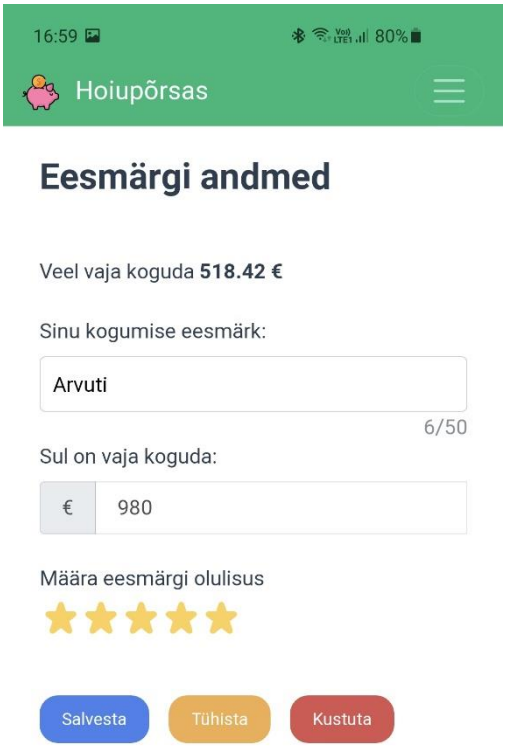
Tehingute otsinguleht ilma vasteteta mobiilis – *SearchTransactions*

Tehingute otsinguleht mobiilis, otsing tehingukirjelduse järgi – *SearchTransactions*

Tehingute otsinguleht mobiilis, otsing kategooria järgi – *SearchTransactions*

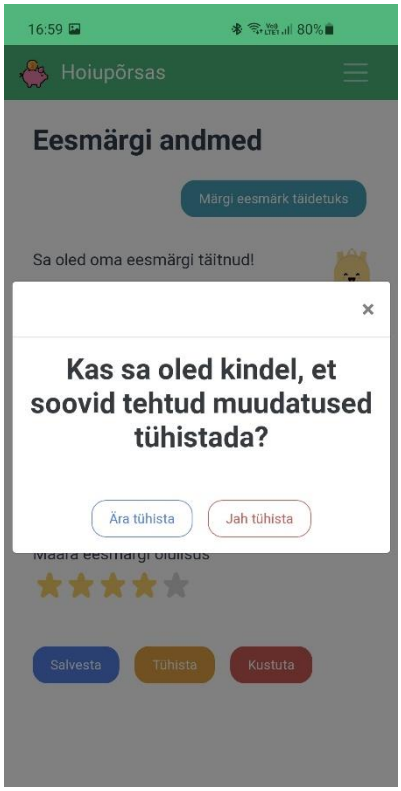


Kogumiseesmärgi lisamisvaade mobiilis – *AddGoal*

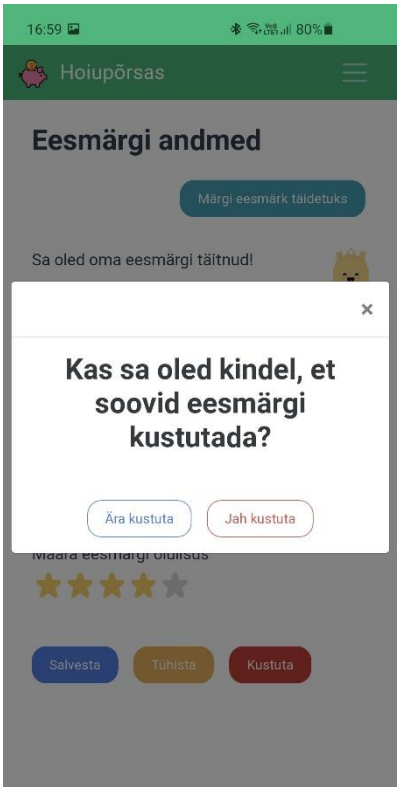


Täitmata kogumiseesmärgi detailvaade mobiilis – *GoalDetail*

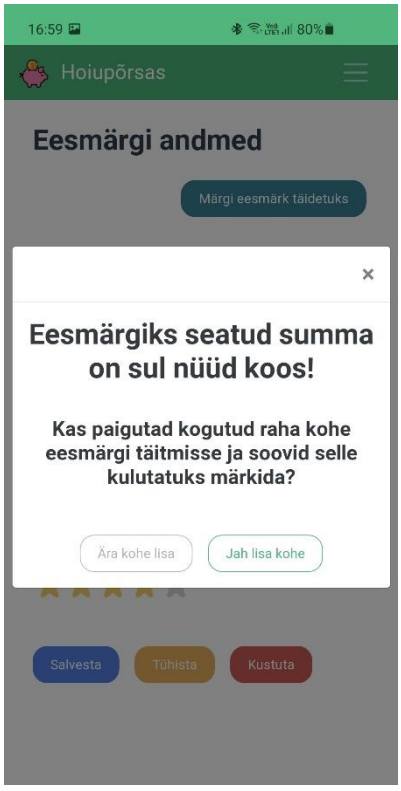
Täidetud kogumiseesmärgi detailvaade mobiilis – *GoalDetail*



Kogumiseesmärgi muudatuste tühistamise kinnitusmodaal mobiilis – *GoalDetail*

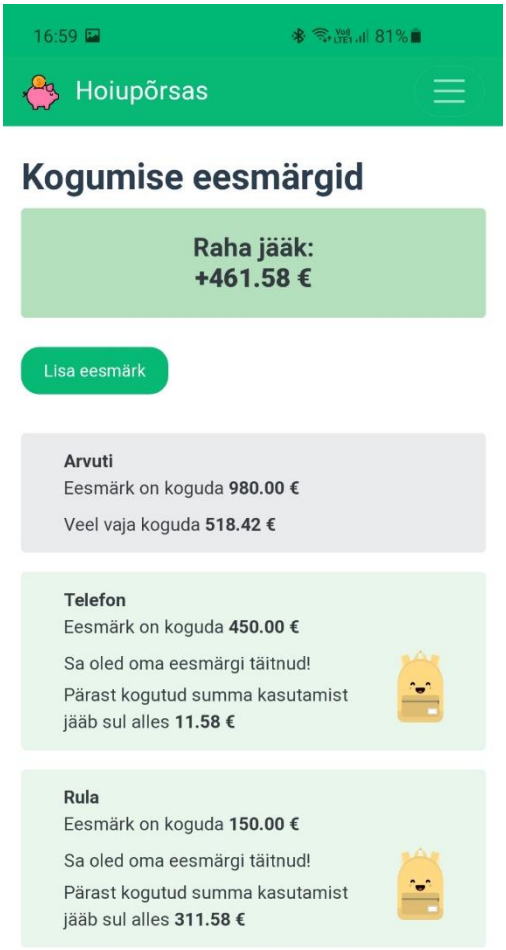


Kogumiseesmärgi kustutamise kinnitusmodaal mobiilis – *GoalDetail*

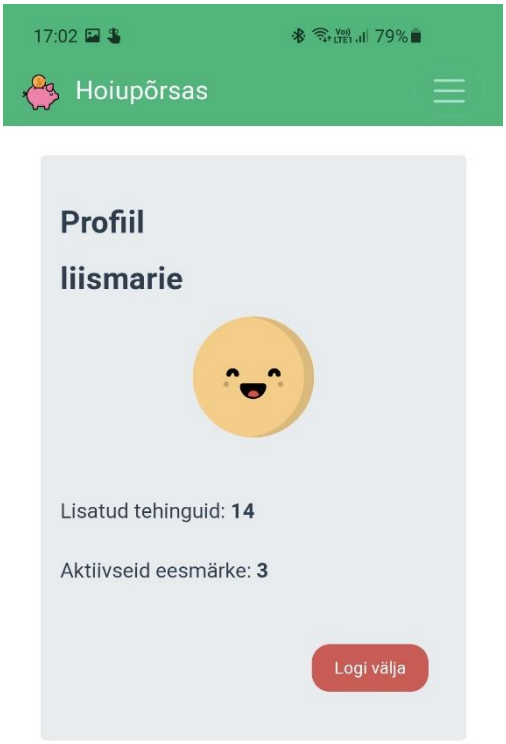


Kogumiseesmärgi täidetuks märkimise modaal mobiilis – *GoalDetail*

Tühi kogumiseesmärkide leht mobiilis – *GoalsOverview*

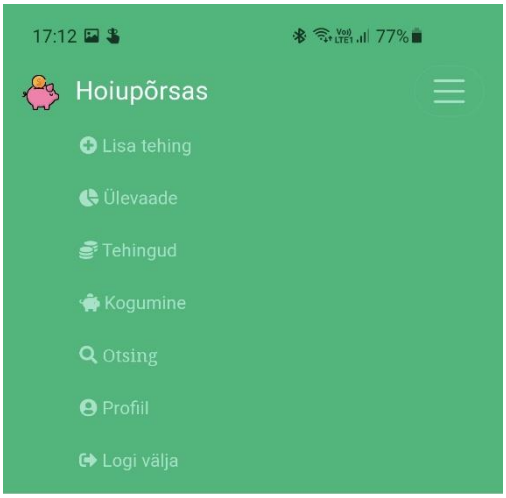


Kogumiseesmärkidega kogumiseesmärkide leht mobiilis – *GoalsOverview*

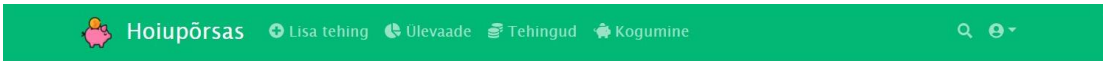


Profiilivaade mobiilis – *Profile*

Otsitud lehte ei leitud vaade mobiilis – *PageNotFound*

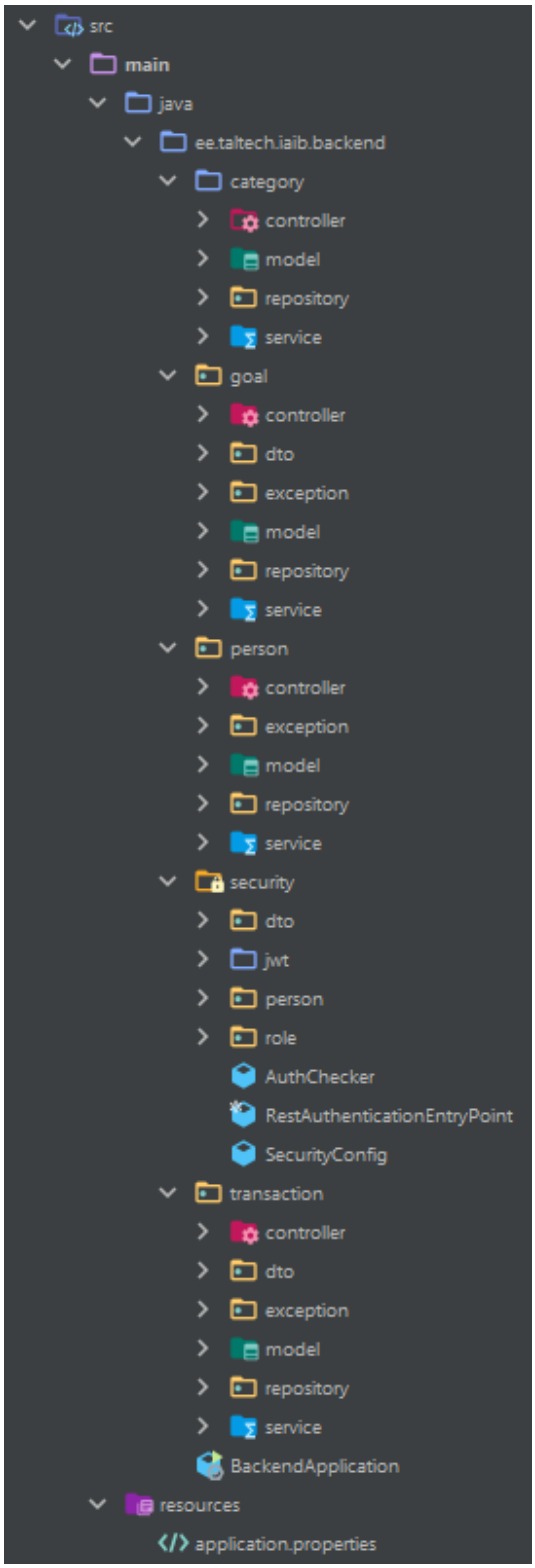


Menüü mobiilis



Menüü arvutis

Lisa 5 – Projekti tagarakenduse arhitektuur



Lisa 6 – Võimalikud API päringud

Kasutajatega seotud API päringud.

Meetod	**Aadress**	**Kirjeldus**	**Parameetrid**
POST	/api/persons/register	Kasutajaks registreerimine	username: String password: String
POST	/api/persons/login	Kasutaja sisselogimine	username: String password: String
POST	/api/persons/logout/{*personUsername*}	Kasutaja väljalogimine	*personUsername*: String

Kogumiseesmärkidega seotud API päringud.

Meetod	**Aadress**	**Kirjeldus**	**Parameetrid**
GET	/api/person/{*personId*}/goals	Tagastab kõik eesmärgid	*personId*: String
GET	/api/person/{*personId*}/goals/amount	Tagastab eesmärkide arvu	*personId*: String
GET	/api/person/{*personId*}/goal/{goalId}	Tagastab eesmärgi	*personId*: String *goalId*: Long
POST	/api/goals	Lisab eesmärgi	goalPersonId: Long goalName: String goalAmount: Double goalPriority: Integer
DELETE	api/goals/person/{*personId*}/goal/{*goalId*}	Kustutab eesmärgi	*personId*: String *goalId*: Long
PUT	/api/goals/{*goalId*}	Muudab eesmärki	*goalId*: Long goalPersonId: Long goalName: String goalAmount: Double goalPriority: Integer

Kategooriatega seotud API päringud.

Meetod	**Aadress**	**Kirjeldus**	**Parameetrid**
GET	/api/categories	Tagastab kõik kategooriad	
GET	/api/categories/income/ {*isIncome*}	Tagastab kas sissetulekute või väljaminekute kategooriad	*isIncome*: Boolean

Tehingutega seotud API päringud.

Meetod	**Aadress**	**Kirjeldus**	**Parameetrid**
GET	api/person/{*personId*}/ transaction-groups/{*time*}	Tagastab tehingud vastavalt perioodile	personId: Long time: all || past || future
GET	/api/person/{*personId*}/ transactions/search	Tagastab tehingud, mis vastavad otsitud parameetrile	*personId*: Long input: String start: String end: String
GET	/api/person/{*personId*}/ transactions/amount	Tagastab tehingute arvu	*personId*: Long
GET	api/person/{*personId*}/ transaction-group/ {*transactionId*}	Tagastab tehingu	*personId*: Long *transactionId*: Long
GET	/api/person/{*personId*}/ transactions/balance	Tagastab kontobalansi	*personId*: Long
POST	/api/transactions	Lisab tehingu	transactionPersonId: Long transactionCategoryId: Long transactionAmount: Double transactionDate: Date transactionDescription: String
DELETE	/api/transactions/person/{ *personId*}/transaction/ {*transactionId*}	Kustutab tehingu	*personId*: Long *transactionId*: Long
PUT	/api/transactions/ {*transactionId*}	Muudab tehingut	*transactionId*: Long

Lisa 7 – Nõuetele vastavus

Funktsionaalsetele nõuetele vastavus.

Nõue	**Implementeeritud**
Rakenduses on võimalik luua kasutajakontot.	Jah
Rakendusse on võimalik logida sisse olemasoleva kasutajana.	Jah
Kasutaja saab vaadata oma profiili.	Jah
Kasutaja saab välja logida.	Jah
Kasutaja näeb ülevaadet oma sissetulekutest ja väljaminekutest.	Jah
Kasutajale kuvatakse sissetulekuid ja väljaminekuid kuupäeva järgi kahanevas järjekorras. Loendvaates kuvatakse ka iga päeva rahalist kokkuvõtet.	Jah
Kasutajale kuvatakse sissetulekuid ja väljaminekuid nädala, kuu, kõik korraga ja kasutaja enda valitud perioodi kaupa, sealjuures kuvatakse ka perioodi rahalist kokkuvõtet.	Jah
Kasutaja saab sissetulekuid ja väljaminekuid otsida kategooria ning kirjelduse järgi.	Jah
Kasutaja saab lisada, muuta ning kustutada sissetulekuid ja väljaminekuid.	Jah
Kasutaja saab sissetulekute ja väljaminekute juures defineerida järgnevad atribuudid: summa, kategooria, kuupäev ja kirjeldus. Nendest atribuutidest kõik, välja arvatud kirjeldus, on kohustuslikud.	Jah
Kasutajale kuvatakse kogumise eesmärke tema määratud olulisuse järjekorras.	Jah
Kasutaja saab lisada, muuta, märkida tehtuks ning kustutada kogumise eesmärke.	Jah
Kasutaja saab kogumise eesmärkide juures defineerida järgnevad atribuudid: summa, kirjeldus ja olulisus. Kõik atribuudid on kohustuslikud.	Jah

Mittefunktsionaalsetele nõuetele vastavus.

Nõue	**Implementeeritud**
Rakenduse kasutajaliides on eestikeelne.	Jah
Rakenduse lähtekood on ingliskeelne.	Jah

Nõue	**Implementeeritud**
Rakendus peab olema kasutatav Google Chrome veebibrauseri värskeimas versioonis.	Jah
Kõik kasutaja poolt saadetud sisestatud andmed peavad olema valideeritud.	Jah
Rakendus peab olema kasutaja jaoks arusaadav ja lihtne kasutada.	Jah
Rakenduse kasutajaliides peab olema ekraani suurusega kohalduv.	Jah
Rakendusele peab olema lisatud PWA võimekus.	Jah
Rakendus peab olema ligipääsetav domeeni kaudu.	Jah

Lisa 8 – Kasutajatestimise tulemused

Kasutusjuht	**Tulemus**
Loo uus kasutaja ja logi sisse	6 testijat said iseseisvalt hakkama.
Vaata oma tehinguid	6 testijat said iseseisvalt hakkama.
Vaata teatud perioodi tehinguid	3 testijat said iseseisvalt hakkama. 3 testijale esitati suunav küsimus „Kas saab muuta kuvatavate tehingute perioodi?“, pärast mida said nad iseseisvalt hakkama.
Otsi tehinguid kategooria ja/või kirjelduse järgi	2 testijat said iseseisvalt hakkama. 4 testijale esitati suunav küsimus „Kas rakendus võimaldab tehinguid otsida?“, pärast mida said nad iseseisvalt hakkama.
Lisa uus tehing	6 testijat said iseseisvalt hakkama.
Muuda tehingut	1 testija sai iseseisvalt hakkama. 5 testijale esitati hüpoteetilise olukorra kirjeldus „Sa panid X tehingusse väära informatsiooni. Mida on nüüd võimalik edasi teha?“, pärast mida said nad iseseisvalt hakkama.
Kustuta tehing	2 testijat said iseseisvalt hakkama. 4 testijale esitati hüpoteetilise olukorra kirjeldus „Sa avastad väära tehingu X. Mida on nüüd võimalik edasi teha?“, pärast mida said nad iseseisvalt hakkama.
Lisa uus eesmärk	6 testijat said iseseisvalt hakkama.
Muuda eesmärki	6 testijale esitati hüpoteetilise olukorra kirjeldus „Sa panid X eesmärki väära informatsiooni. Mida on nüüd võimalik edasi teha?“, pärast mida said nad iseseisvalt hakkama.
Kustuta eesmärk	6 testijale esitati hüpoteetilise olukorra kirjeldus „Sa avastad aegunud eesmärgi X. Mida on nüüd võimalik edasi teha?“, pärast mida said nad iseseisvalt hakkama.
Logi välja	6 testijale esitati hüpoteetilise olukorra kirjeldus „Sa kasutasid oma kasutajat sõbra telefonis. Sõber soovib telefoni tagasi. Kas sa soovid jääda sisse logituks?“, pärast mida said nad iseseisvalt hakkama.

Lisa 9 – Projekti viited

Veebileht	**Aadress**
Rakenduse veebileht	https://hoiuporsas.com/
Staatilise eesrakenduse veebileht	https://piggybank-bd86a.web.app/
Heroku tagarakenduse veebileht	https://piggybankw.herokuapp.com/
Heroku eesrakenduse veebileht	https://piggybank-e.herokuapp.com/
Versioonihalduse veebileht	https://gitlab.cs.ttu.ee/likutt/iaib