TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Aleksei Žavoronkov 193948IADB

Korvpallil baseeruvaid mänge toetav veebirakendus

Bakalaureusetöö

Juhendaja: Meelis Antoi
Magistrikraad

Tallinn 2022

Autorideklaratsioon

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

Autor: Aleksei Žavoronkov

12.05.2022

Annotatsioon

Käesoleva bakalaureusetöö eesmärgiks on luua põhiliselt lastele mõeldud veebirakendus, mis toetab korvpallil baseeruvaid mänge. Veebirakenduse kood on avalik, see võimaldab igaühel anda oma panus projekti arendusse.

Arendusprotsessi käigus luuakse kohanduv veebirakendus, mis võimaldab kasutajatel luua endale konto, otsida ja lisada sõpru, valida korvpallil baseeruvaid mänge, jälgida ja mõjutada mänguprotsesse ning salvestada mängu tulemusi. Arendus koosneb kahest osast: teenusepoolse ja kliendipoolse keskkonna arendamisest.

Rõhk on klientrakenduse funktsionaalsuse teostamisel ja sellel, et klientrakendus oleks kohanduv ja kasutatav erinevate ekraani suurusega seadmetes (mobiil, tahvelarvuti, arvuti). Muuhulgas hõlmab lõputöö kasutatud tehnoloogiate kirjeldust, arendusmustreid, arendusmudeleid ja veebirakenduse turvalisust. Tulemuseks on töötav ja veebis kättesaadav kohanduv veebirakendus.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 28 leheküljel, 6 peatükki, 5 tabelit.

Abstract

A Web Application That Supports Games Based on Basketball

The aim of this bachelor's thesis is to create a responsive web application for children that supports games based on basketball. The web application has open source code. It allows everyone to contribute to the development of the project.

During the development the responsive web application is created. The application allows users to create an account, search and add friends, choose among games, monitor and interact with game processes and save game results. Development consists of two parts: the development of service-side and customer-side applications.

The emphasis in the work is on the implementing the functionality of the client application and on making the client application responsive and usable on devices with different screen sizes (mobile, tablet, computer).

Among other things, the paper covers the technologies used, development patterns, development models and security of the web application. The result is a working and responsive web application.

The thesis is in Estonian and contains 28 pages of text, 6 chapters, 5 tables.

Lühendite ja mõistete sõnastik

API	*Application Programming Interface* - rakendusliides
backend	Teenusepoolne keskkond
Bearer authentication	HTTP autentimisskeem
BPMN	*Business Process Modeling Notation* - ärianalüüsi rakenduste visuaalne modelleerimiskeel
CSS	*Cascading Styling Sheet* – veebilehe küljendamisel kasutatav keel
database context	Põhiklass, mis koordineerib andmemudeli funktsionaalsust (EF Core puhul)
database provider	Pistikprogrammi teek andmebaasiga töötamiseks
DBMS	*Database Management System* - andmehaldussüsteem
DTO	*Data Transfer Object* – programmeerimis keeles loodud klass andmete saatmiseks ja saamiseks
frontend	Kliendipoolne keskkond
hosting	Teenus, mis pakub ressursside paigutamist serverisse, millel on pidevalt juurdepääs võrgule.
HTML	*Hyper Text Markup Language* – veebilehe märgendamise keel
HTTP	*Hyper Text Transfer Protocol* - protokoll teabe edastamiseks arvutivõrkudes
IDE	*Integrated Development Environment* - rakenduste loomiseks mõeldud tarkvara, mis ühendab levinud arendajatööriistad üheks graafiliseks kasutajaliideseks
JWT	*JavaScript Object Notation Web Token* - tööstusstandardi RFC 7519 meetod nõuete turvaliseks esitamiseks kahe osapoole vahel
MVP	*Minimum Viable Product* – minimaalne elujõuline toode
NPM	*Node Package Manager* – pakettide ja teekide haldur
ORM	*Object-Relational Mapping* – tehnika (samuti teek, mis kasutab seda tehnikat), mis võimaldab objektorienteeritud paradigmat kasutades töötada andmebaasiga
REST	*Representational State Transfer* – veebiteenusega suhtlemise liidese arhitektuur

SASS	*Syntactically Awesome Style Sheets* - CSS tehnoloogial põhinev veebilehe küljendamisel kasutatav keel
SCSS	*Sassy Cascading Styling Sheet* – SASS süntaks
SFC	*Single-File Component* - failivorming, mis võimaldab panna Vue,js raamistiku komponendi malli, loogikat ja stiili ühte faili
SPA	*Single Page Application* – üheleherakendus

Sisukord

Tabelite loetelu

1 Sissejuhatus

Korvpalil baseeruvad mängud on algselt loodud selle eesmärgiga, et õpetada algajatele korvpalluritele erinevaid liikumisi, tehnikaid ja põhimõisteid. See on loogiline harjutuste arengukäik, kuna õppimine ja õpetamine läbi mängu on kiirem ja huvitavam võrreldes klassikaliste harjutustega. Tagantjärele saavutasid need mängud populaarsust ka väljaspool korvpalli õpetavaid asutusi, ning lapsed hakkasid mängima neid sama palju kui teisi mänge.

Korvpallil baseeruvad mängud vajavad enamikel juhtudel kahe või enama inimese kaasamist, korvpalliplatsi ja palli. Mängude kestus sõltub mängijate arvust ja keskmiselt kulutatakse umbes 10 minutit ühe mängu peale. Olenevalt geograafilisest asukohast varieeruvad mängude liigid ja nende reeglid. Peamised takistused mängimise jooksul on mänguprotsessi jälgimine ning seisu meelde jätmine (punktide seis, mängijate järjekord, viskekohad).

Käesolev lõputöö tegeleb probleemi analüüsiga Eesti kontekstis ja kasutusele võetakse mängud, mis on enim levinud ning tuntud riigis elavatele harrastajatele. Probleemi pakutud lahenduseks on kohanduv veebirakendus, mis võimaldaks korvpallil baseeruvate mängude mänguprotsesse muuta kergesti jälgitavamateks ja mugavamateks. Lisaks võimaldab veebirakendus vältida vastuvaidlemisi mängijate vahel ning tutvuda uute mängudega.

Lõputöö autor tegeleb professionaalselt korvpalliga rohkem kui 10 aastat ning korraldab spordiüritusi, mille programmidesse kuuluvad ka korvpallil baseeruvad mängud. Autor märkas oma kogemuste põhjal ja ka harrastajate puhul, et varem mainitud takistused segavad mänguprotsessi nautimist. Autor järeldab, et lahendamiseks on vaja luua kohanduv veebirakendus, mis oleks võimeline lahendama eelnimetatud probleeme.

1.1 Metoodika

Antud lõputöö algfaasis selgitatakse lahti eksisteerivaid probleeme ning pakutakse püstitatud nõuetele vastav IT lahendus, mida on võimalik ka edaspidi arendada.

Lahenduse analüüsfaasis valitakse välja sobivaid korvpallimänge ning uuritakse nende mänguprotsesse. Järgmiseks sammuks on rakenduse funktsionaalsete ja mittefunktsionaalsete nõuete väljatoomine. Edasi uuritakse veebirakenduse arhitektuure ning võrreldakse tehnoloogiad, mida on võimalik kasutusele võtta ning põhjendatakse nende valikut.

2 Probleemi ülevaade

Peamine probleem tuleneb sellest, et lapsed viidavad arvuti taga liiga palju aega ja ei pööra vajalikku tähelepanu sportlikule aktiivsusele. Selline suhtumine kahjustab laste tervist ning suurendab haigestumist. Et seda olukorda parandada, on vaja rohkem tegeleda spordiga, kuna füüsilised harjutused ja spordimängud on terve keha, tugeva vaimu ja sotsiaalsete oskuste arendamiseks hädavajalikud [1].

Laps võib tegeleda spordiga erinevates spordiasutustes, näiteks külastada jõusaale, batuudikeskusi, basseine, treeninguid spordikoolide baasil. Kuid kõik need võimalused on tasulised ja laps ei saa iseseisvalt neid valida, kuna vajab lapsevanemate luba ja rahalist toetust. Laps võib aga iseseisvalt ja tasuta sisustada oma vaba aega spordimängudega. Nende läbiviimiseks sobivad koolide spordisaalid, õues asuvad väljakud ja staadionid. Laste spordimängud baseeruvad reeglina erinevatel spordialadel. Probleemi lahendamiseks sobivad hästi korvpallil baseeruvad mängud, kuna nad panevad liikuma mängijate kõiki kehaosi.

Raskuseks on see, et korvpallil baseeruvate mängude protsessid vajavad info säilitamist kõikidel etappidel. Seda kohustust võtavad praegu mängijad enda peale, mis viib olukorrani, kus hakatakse vaidlema, unustatakse järjekord või punktide seis jääb segaseks. Lisaprobleemiks on asjaolu, et korvpallil baseeruvate mängude kirjeldused puuduvad või on raskesti leitavad ja neid levitatakse suuliselt. Seetõttu tuleb iga uue mängu alguses mängureeglid kokku leppida.

2.1 Olemasolevad lahendused

Eksisteerivate lahenduste otsimine näitas, et praegu pole lahendusi, mis toetaks korvpallil baseeruvaid mänge. Suuliselt levinud reeglid ei lahenda püstitatud probleeme. Selline olukord tugevdab autori soovi panustada vastava IT lahenduse arendamisesse.

2.2 Lahenduse skoop

Kuna lahendusi, mis toetaks korvpallil baseeruvaid mänge, ei eksisteeri, siis antud lõputöö autor pakub uue kohanduva veebirakenduse.

Lõputöö kontekstis tegeletakse brauseripõhise lahendusega. See võimaldab kasutada rakendust erinevates seadmetes. Samuti peab veebirakendus olema kohanduv, et tagada sõbralikku ja arusaadavat kasutamist. Eeliseks on see, et brauseripõhine lahendus ei sunni kasutajat lisarakendusi alla laadima vaid vajab ainult interneti ühendust.

Lahendus sisaldab ainult korvpallil baseeruvaid mänge, kuid jätab võimaluse tulevikus lisada ka teistel spordialadel baseeruvaid mänge (võrkpall, jalgpall). Nii on võimalik hõlmata suuremat auditooriumit. Et seda ellu viia, tuleb teha projekt avalikuks, mis võimaldaks huvilistel lähtekoodiga tutvuda ja veebirakendust edasi arendada.

3 Veebirakenduse analüüs

Antud peatükis tehakse korvpallil baseeruvate mängude analüüsid, määratakse nõuded ja valitakse sobilik veebirakenduse arhitektuur. Samuti otsustatakse, millised tehnoloogiad võetakse kasutusele püstitud probleemi lahendamiseks.

3.1 Korvpallil baseeruvate mängude analüüs

Korvpallil baseeruvate mängude loetelu on küllaltki pikk ja lõputöö skoopi kõiki neid mahtuda pole võimalik. Valik kitsendatakse ning võetakse arvesse, et probleemi analüüs viiakse läbi Eesti kontekstis.

Korvpallil baseeruvate mängude valik:

- „-5“
- „33“
- „Around the world”
- „21”

Tuleb teha valitud mängude protsesside analüüs, et oleks võimalik pakkuda kasutajale sobiv kasutajaliides. Lisaks sellele on tehtud otsus koostada mänguprotsesside skeemid BPMN (*Business Process Modeling Notation*) abil, kuna selline visuaalne esitlus võimaldab protsessi loogikat hõlpsasti mõista [2]. Skeemidega saab tutvuda peatükis Lisa 2.

3.1.1 Mäng „-5“

Mängust võtavad osa vähemalt kaks mängijat. Esimene mängija sooritab viset vabalt valitud koha pealt. Kui ta tabas korvi, siis algab uus ring. Selle käigus peab iga mängija tabama korvi sama koha pealt. See, kes viskas oma viske mööda saab „-1“ punkti. Järgmine mängija võib sooritada viske vabalt valitud koha pealt. Kui aga ring on läbi, siis

ringi alustaja võib valida koha järgmise viske sooritamiseks. Kui mängija kogus „-5“ punkti, siis ta lahkub mängust. Võidab see, kes jääb viimasena mängu.

3.1.2 Mäng „33“

Mängust võtavad osa vähemalt kaks mängijat. Esimene mängija sooritab viske vabaviske joone tagant. Kui ta tabas, siis saab ta 3 punkti ja viskab veel kord sama koha pealt. Kui ta viskas mööda, siis teine mängija viskab sealt, kus ta palli püüdis. Kui teine mängija tabas, siis läheb ta vabaviske joone taha. Kui ta pani mööda, siis järgmine mängija viskab sealt, kus ta palli püüdis. Niimoodi käib mäng niikaua, kuni keegi kogub 30 punkti. Kui mängijal on 30 punkti, siis tohib ta sooritada viset ainult kolmepunktijoone tagant. Iga tabamine annab talle 1 punkti. Võidab see, kes esimesena kogub 33 punkti.

3.1.3 Mäng „Around the world“

Mängust võtavad osa vähemalt kaks mängijat ja mängitakse ühel korvpalliväljaku poolel. Mängijad püüavad läbida kõiki ette määratud kohti. Igal mängijal on kolm võimalust sooritada oma viset:

- Vise nimega „Vaba“ – igal mängijal on õigus sooritada vaba vise iga koha pealt ilma karistuseta möödaviskamise eest. Järgmine vise toimub kohalt, kus oli viimane möödavise.

- Vise nimega „Võimalus“ – igal mängijal on õigus sooritada teine vise iga koha pealt. Kui see vise läheb mööda, siis mängija peab alustama oma mängu algusest.

- Vise nimega „Elu“ – igal mängijal on õigus sooritada kolmas vise iga koha pealt. Kui see vise läheb mööda, siis mängija kaotas ja lahkub mängust.

Esimene mängija alustab esimeselt ette määratud kohalt ja liigub järgmisele kohale iga tabatud viskega. Pärast möödaviskamist peab ta andma palli järgmisele mängijale või valima vastavalt olukorrale kas viske nimega „Võimalus“ või viske nimega „Elu“. Võidab see mängija, kes esimesena läbib kõik ette määratud kohad [3].

3.1.4 Mäng „21“

Mängust võtavad osa vähemalt kaks mängijat. Esimene mängija sooritab viske vabaviskejoone tagant. Kui ta tabas korvi, saab ta 1 punkti. Pärast korvi tabamist läheb mängija alati vabaviskejoone taha ja teeb oma järgmise viske sealt. Kui ta paneb mööda,

siis järgmine mängija püüab palli ja teeb oma viske sealt, kus ta palli püüdis. Kui enne püüdmist pall maha ei kukkunud, siis saab mängija tabamisel 2 punkti. Kui enne viset pall põrkas maast, siis saab mängija tabamisel 1 punkti. Kui mängija viskab mööda, siis järgmine mängija püüab palli ja sooritab oma viske. Nii käib mäng momendini, kui keegi mängijatest kogub 21 punkti. Kui mängijal on 10 või 20 punkti, ning ta viskab mööda, kaotab ta kõik oma kogutud punktid ja alustab otsast peale.

3.2 Nõuete määramine

Nõuete määramine on tehtud kasutajalugude abil. Põhirollis on korvpallil baseeruvate mängude mängija (edaspidi tavakasutaja).

3.2.1 Funktsionaalsed nõuded

Kontopõhised funktsionaalsed nõuded:

- Tavakasutajana soovin registreeruda.
- Tavakasutajana soovin oma kontot kustutada.
- Tavakasutajana soovin sisse logida.
- Tavakasutajana soovin välja logida.
- Tavakasutajana soovin eelnevalt mängitud mänge vaadata.
- Tavakasutajana soovin eelnevalt mängitud mänge kustutada.
- Tavakasutajana soovin otsida sõpru fraasi järgi.
- Tavakasutajana soovin lisada sõpru.
- Tavakasutajana soovin kustutada sõpru.

Mängupõhised funktsionaalsed nõuded:

- Tavakasutajana soovin lisada mängule mängijaid kirjutades ainult nende nimesid.
- Tavakasutajana soovin lisada mängule oma sõpru.

- Tavakasutajana soovin eemaldada mängust mängijaid.
- Tavakasutajana soovin valida mitmest mängust.
- Tavakasutajana soovin mängu kirjeldust vaadata.
- Tavakasutajana soovin mängu jooksul vahetulemusi vaadata.
- Tavakasutajana soovin mängu lõpetada.
- Tavakasutajana soovin mängu salvestada.

Mäng „-5“:

- Tavakasutajana soovin vähendada mängija punkte ühe võrra.
- Tavakasutajana soovin näha, kas „ring“ käib.
- Tavakasutajana soovin määrata, et mängija viskas mööda.

Mäng „33“:

- Tavakasutajana soovin lisada mängijale 3 punkti.
- Tavakasutajana soovin lisada mängijale 1 punkti.
- Tavakasutajana soovin määrata, et mängija viskas mööda.

Mäng „21“:

- Tavakasutajana soovin lisada mängijale 2 punkti.
- Tavakasutajana soovin lisada mängijale 1 punkti.
- Tavakasutajana soovin määrata, et mängija viskas mööda.

Mäng „Around the world“:

- Tavakasutajana soovin lisada mängijale 1 punkti.
- Tavakasutajana soovin näha, kust pean viskama.

- Tavakasutajana soovin näha, milline on praeguse viske tüüp.
- Tavakasutajana soovin määrata, et mängija viskas mööda.

3.2.2 Mittefunktsionaalsed nõuded

- Tavakasutajana soovin kasutada veebirakendust mobiili peal.
- Tavakasutajana soovin kasutada veebirakendust tahvelarvuti peal.
- Tavakasutajana soovin kasutada veebirakendust arvuti peal.
- Tavakasutajana soovin kasutada veebirakendust ilma konto registreerimiseta.

3.3 Veebirakenduse arhitektuur

Tänapäeval on olemas kaks põhilist lähenemist veebirakenduse ehitamisele. Esimene on traditsiooniline, kus suurem osa äriloogikast toimub serveris. Teine on SPA (*Single Page Application*), kus suurem osa kasutajaliidese loogikast toimub brauseris ja suhtlemine serveriga toimub läbi veebi API (*Application Programming Interface*).

Traditsioonilist lähenemisviisi kasutatakse sellistes olukordades:

- Veebirakenduse kliendipoolne funktsionaalsus ei oma mingit keerukust või pakub andmeid vaatamiseks.
- Veebirakendus peab töötama brauseris ilma JavaScript toetuseta.

SPA lähenemisviis sobib sellistel juhtudel:

- Kliendipoolne funktsionaalsus on rikkas või keeruline.
- Arendaja tunneb piisavalt JavaScript programmeerimiskeelt.
- Veebirakendus peab avaldama API teistele kasutajaliidestele [4].

Kuna planeeritava veebirakenduse funktsionaalsus on keerulisem kui lihtsalt andmete vaatamine ning suurem osa kasutajaliidese loogikast peab toimuma brauseris, siis kasutusele võetakse SPA lähenemisviis. Samuti on selle otsuse kasuks asjaolu, et antud lõputöö autor tunneb piisavalt JavaScript programmeerimiskeelt.

3.4 Tehnoloogiate valik

Sobiva tehnoloogia valimine on keeruline. Ei ole universaalset lahendust, mis aitaks seda otsust teha. Loogiline on valida tehnoloogia, millega on mugavam töötada ja mõista kõiki eripärasusi, eeliseid ja puudusi. Samuti peab olema tehnoloogial hea tugi. Antud lõputöö raames püütakse luua MVP (*Minimum Viable Product*). Seepärast ei ole soovitav takerduda probleemidesse, mis võivad olla põhjendatud tehnoloogia nooruse või vaese dokumentatsiooniga [5]. Veebirakenduse loomiseks on vaja valida *backend* programmeerimiskeel, *backend* raamistik, andmebaas, *frontend* raamistik, IDE (*Integrated Development Environment*).

2021. aastal koostas ülemaailmne tarkvaramüüja JetBrains aruande „The State of Developer Ecosystem 2021“, kus on kirjeldatud tehnikatööstuse suundumusi. Aruande koostamisel kasutati 31,743 arendaja arvamust 183 riigist või piirkonnast [6]. Tehnoloogiate võrdlemiseks kasutatakse andmeid sellest aruandest. Kõik järgmistes peatükkides vaadatud tehnoloogiad on enam levinud ning hea dokumentatsiooniga. Seepärast puuduvad tabelites veerud vastavate nimedega. Jääb ainult üks võrdlemise aspekt – mugavus. Mugavuse all mõistab lõputöö autor tehnoloogia kättesaadavust, võttes arvesse ka enda kogemust.

3.4.1 *Backend* tehnoloogiate valik

Tabel 1 ja Tabel 2 esitavad sobivad *backend* tehnoloogiad. Antud lõputöö raames valitakse programmeerimiskeeleks C# ja raamistikuks ASP.NET Core.

Tabel 1.Teenusepoolse keskkonna programmeerimiskeelte võrdlemine

Nimi	Kättesaadavus	Kogemus
Python [6]	Hea [7]	Keskmine
JavaScript [6]	Hea [8]	Keskmine
Java [6]	Hea [9]	Keskmine
C# [6]	Hea [10]	Hea
PHP [6]	Hea [11]	Madal

Tabel 2.Teenusepoolse keskkonna raamistike võrdlemine.

Nimi (keel)	Kättesaadavus	Kogemus
Flask (Python) [12]	Hea [17]	Puudub
Django (Python) [12]	Hea [18]	Puudub
Express (JavaScript) [13]	Hea [19]	Puudub
Spring Boot (Java) [14]	Hea [20]	Madal
ASP.NET Core (C#) [15]	Hea [21]	Hea
Laravel (PHP) [16]	Hea [22]	Puudub

3.4.2 Andmebaasi valik

Andmete hoidmiseks on vaja andmebaasi. ASP.NET Core töötab andmebaasiga kasutades Entity Framework Core ORM (*Object-Relational Mapping*) [23].

Entity Framework Core (edasi EF Core) toetab mitmeid erinevaid andmebaasisüsteeme. Seega on võimalik töötada mis tahes DBMS-iga (*Database Management System*), kui selleks on olemas vajalik *database provider*. Vaikimisi pakutakse praegu mitmeid sisseehitatud *database providers*: MS SQL Server, SQLite, PostgreSQL. Saadaval on ka kolmanda osapoole pakkujad, näiteks MySQL, MariaDB. Samuti väärib märkimist, et EF Core pakub andmetega töötamiseks üldist API-t. See tähendab, et kui otsustatakse DBMS-i muuta, siis puudutavad projekti peamised muudatused eelkõige konfiguratsiooni ja ühenduse seadistusi. Kood, mis töötab otseselt andmetega, jääb samaks [23].

Kõige sagedamini kasutavad ASP.NET Core raamistikuga tehtud veebirakendused MS SQL Server andmebaasi [24]. Antud lõputöö raames pannakse rõhk arenduse mugavusele ja kiirusele. Seepärast otsustakse võtta MS SQL Server andmebaasi asemel SQLite andmebaas, kuna see on väike, kiire, iseseisev, suure töökindluse ja

täisfunktsionaalsusega andmebaas [25]. Tulevikus (see jääb lõputöö skoobist välja) on võimalik siis otsustada, milline andmebaas asendab SQLite.

3.4.3 *Frontend* tehnoloogiate valik

Klientrakenduse ehitamine vajab kolm tehnoloogiat: HTML (*Hyper Text Markup Language*), CSS (*Cascading Styling Sheet*) ja JavaScript. HTML kasutatakse veebilehe sisu eraldamiseks, CSS selle sisu paigutamiseks ja stiliseerimiseks ning JavaScript veebilehe käitumise määramiseks [26]. On võimalik luua klientrakendus kasutades otseselt neid tehnoloogiaid. Kuna oodatud veebirakenduse funktsionaalsus omab keskmist keerukust, siis selline lähenemine ei sobi. Mida suuremaks läheb rakendus, seda raskemaks muutub koodi kirjutamine. Et seda probleemi lahendada, tuleb kasutusele võtta SPA raamistik. Tabel 3 esitab kolm kõige levinumat SPA raamistikku [13]. Antud lõputöö raames sobib kõige paremini Vue.js raamistik.

Tabel 3. Kliendipoolse keskkonna raamistike võrdlemine.

Nimi	Kättesaadavus	Kogemus
React [13]	Hea [27]	Keskmine
Vue.js [13]	Hea [28]	Hea
Angular [13]	Hea [29]	Madal

3.4.4 IDE valik

IDE võimaldab kiiresti luua uusi rakendusi, kuna kõik vajalikud tööriistad on esindatud ühes kohas. Nende seadistamine on lihtne ning õppimise aeg on väiksem, võrreldes olukorraga, kus õpitakse kasutama iga tööriista eraldi. IDE funktsionaalsus aitab töövoogu korraldada ja probleeme lahendada. IDE kontrollib ja jälgib programmeerija tegevusi ja teavitab visuaalselt, missuguseid toiminguid on võimalik ette võtta. Sellise kontrolli peaeesmärk on vältida inimfaktoriga põhjustatud vigu. Samuti on võimalik kõiki arendusega seotud toiminguid teha ilma rakendusi vahetamata. See kiirendab oluliselt tööprotsessi. Rakendusi on võimalik arendada ka viisil, kui käsitsi seadistatakse kõiki vajalikke tööriistu. Mõne arendaja jaoks on selle lähenemisviisi eeliseks ülim

kohandamine. Aga reeglina on arenduse kontekstis tähtsamad ajasäästmine, automatiseerimine ja lihtsustamine. Samuti on tänapäevane IDE piisavalt paindlik ja igaüks võib seda seadistada nii, nagu tal on vaja. Seega küsimus pole selles, kas võtta kasutusele IDE, vaid pigem selles, milline IDE valida [30].

Tabel 4 ja Tabel 5 esitavad sobivad IDE-d. Teenusepoolse keskkonna arendamiseks võetakse Visual Studio ja kliendipoolse - Visual Studio Code.

Tabel 4. Teenusepoolse keskkonna arendamiseks IDE võrdlemine.

Nimi	Kättesaadavus	Kogemus
Visual Studio [15]	Hea [31]	Hea
Rider [15]	Halb [32]	Hea
Visual Studio Code [15]	Hea [33]	Hea

Tabel 5. Kliendipoolse keskkonna arendamiseks IDE võrdlemine.

Nimi	Kättesaadavus	Kogemus
Visual Studio Code [13]	Hea [33]	Hea
WebStorm [13]	Halb [34]	Madal

3.5 Analüüsi kokkuvõte

Veebirakenduse analüüsi käigus valitakse neli Eestis enam levinud ja tuntumat korvpallil baseeruvat mängu: „33“, „21“, „-5“ ja „Around the world“. Samuti tehakse nende mänguprotsesside analüüsid, mis annavad paremat ülevaadet nende mängude kohta.

Analüüsis määratakse funktsionaalsed ja mittefunktsionaalsed nõuded. Nõuete määramine tehakse kasutajalugude abil.

Veebirakenduse arhitektuuris võetakse kasutusele SPA lähenemisviis, kuna eeldatav tulemus omab keskmist kliendipoolset keerukust.

Serveripoolsetest keeltest valitakse C# ning serveripooltest raamistikest ASP.NET Core. Sellise otsuse põhjuseks on antud tehnoloogiate põhjalik dokumentatsioon, hea kättesaadavus, võimekus, mugavus ja ka antud lõputöö autori eelnev kogemus.

Andmebaasiks kasutatakse SQLite, kuna see on väike, kiire ja iseseisev täisfunktsionaalsusega andmebaas. Need parameetrid toetavad veebirakenduse arendamist.

Klientrakenduse ehitamiseks võetakse kasutusele Vue.js raamistik, mis võimaldab mugavalt ja korralikult luua nii madala kui ka kõrgema keerukusega SPA.

Antud lõputöö raames kasutusele võetud IDE: Visual Studio Code ja Visual Studio. Mõlemad on hea dokumentatsiooniga, enam levinud ning neid kasutatakse sarnaste tehnoloogiate valikuga.

4 Veebirakenduse arendus

Veebirakenduse arendus koosneb kahest osast. Esimeses osas luuakse teenusepoolne keskkond, teises kliendipoolne keskkond. Järgnevates peatükkides on lähemalt kirjeldatud nende osade arendust.

4.1 Teenusepoolne keskkond

Veebirakenduse *backend* on ehitatud ASP.NET Core raamistiku peale. See vastutab rakenduse äriloogika eest, edastab ja võtab vastu andmeid, salvestab ja loeb andmeid andmebaasist ning tegeleb kasutajate autentimise ja autoriseerimisega.

4.1.1 Arendusmudel

ASP.NET Core raamistik võimaldab luua veebirakendusi, kasutades erinevaid arendusmudeleid. Püstitatud probleemi lahendamiseks võetakse kasutusele ASP.NET Core Web API arendusmudel (Lisa 3). See pakub REST (*Representational State Transfer*) mustri juurutamist, kus igale HTTP (*Hyper Text Transfer Protocol*) päringu tüübile määratakse eraldi ressurss. Sellised ressursid on määratletud kui API kontrolleri meetodid. See mudel sobib eriti hästi SPA jaoks [35].

4.1.2 Arhitektuur

Traditsiooniliselt veebirakenduse teenusepoolses keskkonnas on kolm kihti (Lisa 4).

- Kasutajaliidese kiht – läbi selle kihi teeb kasutaja oma päringuid ja saab vastuseid.
- Äriloogika kiht – selles kihis asub kogu rakenduse loogika.
- Andmete juurdepääsu kiht – selles kihis toimub suhtlemine andmebaasiga ja teiste andmeallikatega.

Kasutajaliidese kiht suhtleb ainult äriloogika kihiga. Äriloogika kiht küsib andmeid andmete juurdepääsu kihist. Andmete juurdepääsu kiht on ainukene, mis suhtleb andmete

allikatega. Igal kihil on omad ülesanded ja nende alusel on võimalik koodi organiseerida [36].

Antud lahenduse raames võetakse traditsiooniline veebirakenduse arhitektuur, aga ilma äriloogika kihita. Selline otsus on tehtud, kuna antud lahenduses puudub keeruline või isegi keskmine äriloogika. *Backend* edastab ja võtab vastu andmeid, salvestab ja loeb andmeid andmebaasist ning tegeleb kasutajate autentimise ja autoriseerimisega. Mitte ükski nendest funktsionaalsustest ei vaja äriloogika kihti. Vaatamata sellele on vaja korralikult koostada kaks ülejäänud kihti, et vajaduse korral oleks kerge lisada äriloogika kiht.

4.1.3 Andmebaas

Analüüsi käigus selgus, et andmebaasiks võetakse kasutusele SQLite, mida on kerge tööle panna. Selleks on vaja paigutada paar teeki kasutades NuGet paketihaldussüsteemi ja määrata ühendus *appsettings.json* failis.

Edasi on siis vaja luua domeenimudelid ning määrata suhted nende vahel. Kuna seda kõike tehakse otseselt koodis, siis on mõistlik eelnevalt koostada andmebaasi skeem.

Kui domeenimudelid on valmis, siis luuakse *database context* klass. Sinna pannakse kirja kõik domeenimudelid ning kirjeldatakse lisakäitumine, mida EF Core peab teostama. Järgmine samm on *Program.cs* failis *database context* klassi registreerimine.

Pärast rakenduse käivitamist luuakse automaatselt andmebaas. Vajaduse korral on võimalik vaadata andmebaasi struktuuri kasutades kolmanda osapoole tarkvara.

4.1.4 Repository ja Unit of Work arendusmustrid

Andmete juurdepääsu kihis toimub suhtlemine andmebaasiga. Et seda suhtlemist lihtsamaks ja mugavamaks teha, võetakse kasutusele kaks arendusmustrit: Repository (Lisa 5) ja Unit of Work (Lisa 6). Nad on mõeldud abstraktsioonikihi loomiseks andmete juurdepääsu kihi ja rakenduse äriloogika kihi vahele (antud lõputöö raames andmete juurdepääsu kihi ja kasutajaliidese kihi vahele) [37].

Iga domeenimudeli jaoks luuakse oma Repository klass. Unit of Work klass koordineerib Repository klasside tööd, luues ühe *database context* klassi, mida iga Repository klass saab kasutada. Kasutajaliidese kiht saab kasutada vajalikku domeenimudelit läbi selle

Unit of Work klassi. Selline lähenemine lihtsustab koodi kirjutamist ja võimaldab tulevikus teha ühiktestimist [37].

4.1.5 Turvalisus

Kõige elementaarsemal tasemel hõlmab turvalisus tagamist, et on teada, kellelt konkreetne taotlus pärineb. Seejärel ka selle tagamist, et kasutajal on olemas juurdepääs ressurssidele, mida ta soovib kasutada [38]. Teiste sõnadega, turvalisuse raames on vaja rääkida autentimisest ja autoriseerimisest.

- Autentimine - kasutaja määratlemise protsess (vastus küsimusele „Kes on kasutaja?“) [39].

- Autoriseerimine - kontrollimise protsess, kas kasutajal on õigus mõnele ressursile ligi pääseda (vastus küsimusele „Millised õigused on kasutajal?“) [39].

Antud lahenduses võetakse kasutusele ainult autentimine, kuna autoriseerimine tähendab kasutajate rollide lisamist. See funktsionaalsus pole antud lõputöö skoobis vajalik. Tuleb samuti mainida, et ASP.NET Core raamistikku on autentimine ja autoriseerimine juba sisse ehitatud.

Autentimine nõuab autentimisskeemi seadistamist ja antud lahenduses kasutatakse *Bearer authentication*. Autentimisskeem võimaldab valida konkreetset autentimistöötlejat, kes teostab kasutajate otsest autentimist päringuandmete alusel. Kui kasutatakse *Bearer authentication*, siis tähendab see, et autentimiseks kasutatakse JWT (*JavaScript Object Notation Web Token*). Väärib märkimist, et JWT abil autentimiseks tuleb projekti lisada vastav teek [39].

4.2 Kliendipoolne keskkond

Kliendipoolne keskkond on loodud Vue.js raamistiku abil. See tugineb standardsetele HTML, CSS ja JavaScript tehnoloogiatele ning pakub deklaratiivset ja komponendipõhist programmeerimismudelit, mis aitab tõhusalt arendada kasutajaliideseid, olgu see siis lihtne või keeruline [40].

4.2.1 Vaated

Vue.js raamistikul on mitu kasutusvõimalust. Üks nendest on SPA loomine [41]. Alguses on vaja planeerida, milliseid vaateid peab rakendus omama. Iga vaade on rakenduse sektsioon, mis vastutab mingi valdkonna funktsionaalsuse eest (Lisa 7). Liikumist klientrakenduse vaadete vahel teostab Vue.js raamistiku ametlik ruuter Vue Router, mis integreerub sügavalt raamistiku tuumaga [42].

Kui vaate loogika keerukus on keskmisel või isegi kõrgel tasemel, siis ainult vaadete kasutamisest ei piisa. Selleks, et keerulist käitumist oleks kerge lisada ja tulevikus ka laiendada või muuta, on vaja jagada vaate funktsionaalsust loogilisteks osadeks. See tähendab, et on vaja luua komponente, mis hakkavad vastutama selle käitumise eest.

4.2.2 Komponendid

Vue.js raamistik pakub kasutamiseks SFC (*Single-File Component*), mis on klassikalise HTML, CSS ja JavaScripti tehnoloogiate loomulik laiendus. SFC on soovitatav kasutamiseks SPA arendamise puhul. SFC tuleb eelnevalt kompileerida JavaScript ja CSS koodiks. Kompileeritud SFC on standardne JavaScript moodul, ning seda on võimalik õige ehituse seadistusega importida [43].

SFC kasutamise eesmärk on parandada koodi hooldatavust. Kui eraldada probleeme failitüüpide eraldamise kaudu, siis ei aita see saavutada eesmärki üha keerulisemaks muutuvate kasutajaliidese rakenduste kontekstis. Kaasaegses kasutajaliidese arenduses on koodi jagamise asemel kolmeks üksteisega põimuvaks kihiks palju mõttekam jagada neid lõdvalt seotud komponentideks. Komponendi sees on selle mall, loogika ja stiilid olemuslikult seotud ning nende paigutamine muudab komponendi sidusamaks ja hooldatavamaks [43].

4.2.3 SASS ja SCSS kasutus

SASS (*Syntactically Awesome Style Sheets*) on veebilehe küljendamisel kasutatav keel, mida kompileeritakse CSS keelde. SASS võimaldab kasutada lisafunktsionaalsust, mis on täielikult ühilduv CSS keele süntaksiga. SASS aitab hoida suuri stiililehti hästi organiseerituna ja hõlbustab disaini jagamist nii projektide sees kui ka nende vahel [44]. Stiililehtede kirjutamisel kasutatakse SCSS (*Sassy Cascading Styling Sheet*), mis on üks SASS süntaksitest.

Antud lahenduse raames on stiililehtede organiseerimine selline:

- Igal vaatel on olemas stiilileht.
- Igal komponendil on olemas stiilileht.
- Kogu rakendusel on olemas üldine stiilileht.

Selline struktuur võimaldab alguses määrata põhistiili reeglid ja siis, vajaduse korral, täiendada neid reegleid nii vaate, kui ka komponendi tasemel.

4.2.4 Suhtlus teenusepoolse keskkonnaga

Teenusepoolse keskkonnaga suhtlemine on teostatud Axios HTTP kliendi abil, mida saab alla laadida NPM (*Node Package Manager*) kaudu. Kuna kõik klientrakenduse HTTP päringud teostatakse samamoodi, siis on vaja luua üks klass, kus on olemas kõik vajalikud meetodid. Pärast on võimalik seda klassi taaskasutada. Klassile on vaja anda veebiaadress, kuhu ta soovib pöörduda. Vajadusel antakse ka DTO (*Data Transfer Object*) andmetega.

4.2.5 Turvalisus

Teenusepoolseks keskkonnaks valitakse *Bearer authentication* autentimisskeem. See tähendab, et sisse logimise õnnestumise puhul saadetakse klientrakendusele JWT. JWT tuleb saata koos iga järgmise HTTP päringuga, et teenusepoolses keskkonnas oleks võimalik kasutajat identifitseerida. Et seda teostada, tuleb kuskil hoida JWT. Selleks on võetud kasutusele olekuhaldusmuster ja teek Vuex, mis töötab terve rakenduse tsentraliseeritud hoidlana [45].

5 Tulemuste kirjeldamine

Suurem osa püstitatud eesmärkidest sai täidetud. Järgmiseks sammuks on veebirakendusele hinnangu andmine ja testimine. Kõige paremini saavad selle ülesandega hakkama sihtgruppi kasutajad. Selleks tehakse klientrakendus kättesaadavaks, kasutades Netlify *hosting* ettevõtte teenust. Netlify pakub tasuta tariifi, mis hästi sobib antud lõputöö skoobi raames.

5.1 Sihtgruppi kasutajate hinnang

Üldiselt on laste hinnang positiivne ning nad kavatsevad ka edaspidi rakendust kasutada. Samuti sai antud lõputöö autori tähelepanu pööratud kohtadele, kus on vaja lisatööd teha.

Kõige rohkem palutakse lisada rakendusele teisi keeli, kuna praegune kasutajaliides toetab ainult inglise keelt. Selline otsus tehti sellepärast, et Eestis elavad erinevatest rahvustest inimesed, kes räägivad erinevaid keeli. Inglise keel on aga rahvusvaheline suhtluskeel, seepärast on ta valitud veebirakenduse keeleks.

Järgmine märkus on mänguprotsessi sündmuste kohta. Mängijad soovivad, et kasutajaliides pakuks võimalust viimast sündmust tagasi võtta (näiteks mängijate vahetamine, punktide lisamine). Selle funktsionaalsuse lisamine pole keeruline ja see tuleb kindlasti ka ära teha. Siiski jääb lisamine lõputöö skoobist väljaspoole.

Mõned kasutajad testisid veebirakenduse täisversiooni. Nendel oli võimalus kasutada ka oma kontoga seotud funktsionaalsust. Selliste kasutajate hinnang on väga kõrge, ning nad soovivad tulevikus kasutada just täis funktsionaalsusega veebirakendust. Selle teostamiseks on vaja siis läbi mõelda, kust võtta ressursse teenusepoolse keskkonna paigutamiseks.

5.2 Antud lõputöö autori hinnang

Enamus funktsionaalseid nõudeid said arenduse ajal valmis. Nii teenusepoolse keskkonna, kui ka kliendipoolse keskkonna loomine oli keskmise keerukusega, ning

jõukohane ja õpetlik. Töö käigus tekkisid takistused koodi organiseerimisel ja programmide arhitektuuri koostamisel. Seda võib põhjendada lõputöö autori vähese kogemusega.

Üks funktsionaalne nõue ei jõudnud valmis: tavakasutajana soovin oma kontot kustutada. Praegune lahendus võimaldab kontot kustutada ainult sellel juhul, kui kasutaja ei jõudnud mingit infot salvestada oma kontos (kui kasutajal ei ole ühtegi salvestatud mängu, kasutajal ei ole ühtegi lisatud sõpra, mitte keegi teistest kasutajatest ei lisanud antud kasutajat sõpradesse). Vastasel juhul konto kustutamine pole võimalik, kuna see otseselt mõjutab palju andmebaasis olevat infot. See on asjaolu, mis vajab lisamõtlemist.

Veebirakendust saab kasutada ka ilma konto registreerimiseta. Siis aga tuleb leppida sellega, et mõni funktsionaalsus on kättesaamatu:

- Kasutaja ei saa kasutada sõpradega seotud funktsionaalsust.
- Kasutaja peab käsitsi sisestama mängijate nimed.
- Kasutaja ei saa salvestada mänge.
- Kasutaja ei saa vaadata oma statistikat.

Vaatamata sellele lahendab veebirakendus püstitatud probleeme.

Kõik mittefunktsionaalsed nõuded said arenduse ajal valmis. Kasutaja võib kasutada veebirakendust mobiili, tahvelarvuti ja arvuti peal. Kõik veebirakenduse vaated ja komponendid muudavad oma suurust vastavalt ekraani suurusele. Lahendus vajab ka täiendamist. Mida suurem ekraan, seda rohkem objekte seal võib olla (mõistlikes kogustes). See tähendab, et vaated võivad erineda kuvatud andmete koguse poolest.

5.3 Edasised sammud

Pärast tagasiside saamist koostati edasiste sammude loetelu. Need sammud jäävad antud lõputöö skoobist välja.

Tagasiside baasil on koostatud edasised sammud:

- Klientrakendusele lisafunktsionaalsuse lisamine.

- Teiste keelte toetuse lisamine.
- Teenusepoolse keskkonna paigutamine (realiseerimiseks on vaja lisaressursse, antud juhul raha).

Lõputöö autori arvamuse baasil koostatud edasised sammud:

- Versioonihaldusele kirjeldamise lisamine.
- Konto kustutamise funktsionaalsuse lisamine.
- Vaadete ja komponentide paigutuse planeerimine klientrakenduses (erinevate seadmete peal peab kasutajakogemus olema maksimaalset kõrgel tasemel).
- Teistel spordialadel (jalgpall ja võrkpall) baseeruvate mängude lisamine.

6 Kokkuvõte

Antud diplomitöö eesmärk oli luua kohanduv veebirakendus, mis toetab korvpallil baseeruvaid mänge.

Püstitatud probleemi analüüs andis hea ülevaate veebirakenduse funktsionaalsetest ja mittefunktsionaalsetest nõutest, korvpallil baseeruvate mängude reeglitest ja mänguprotsessidest ning tehnoloogiatest, mida oli võimalik kasutusele võtta.

Lõputöö arenduskäiku kirjeldav peatükk annab ülevaate teenusepoolse keskkonna arendusmudelist, arhitektuurist, arendusmustritest ja turvalisusest. Samas peatükis kirjeldatakse kliendipoolse keskkonna arhitektuuri, kasutatud tehnoloogiaid, turvalisust ja suhtlust teenusepoolse keskkonnaga.

Tulemuste kirjeldamise peatükis antakse hinnang loodud veebirakenduse funktsionaalsusele. Kirjeldatakse sihtgrupi kasutajatelt saadud tagasisidet, ning koostatakse edasiste sammude loetelu.

Projektile võib anda positiivse hinnangu, kuna enamus funktsionaalsusest sai täidetud arenduse käigus. Seda hinnangut tugevdab ka sihtgrupi kasutajate positiivne tagasiside. Vaatamata sellele on olemas puudused, mida on vaja parandada. Puuduste baasil on koostatud edasiste sammude loetelu.

Projekti kood on avalik (Lisa 8, Lisa 9, Lisa 10), nii teenusepoolse kui ka kliendipoolse keskkonna puhul, seega võib iga soovija anda oma panuse projekti arendusele.

Kasutatud kirjandus

[1] Francis P. Crawley, Peter Hoyer, Artur Mazur, et al., „Health, integrity, and doping in sports for children and young adults. A resolution of the European Academy of Pediatrics,“ *European Journal of Pediatrics*, vol. 176, pp. 825-828, 2017, doi: 10.1007/s00431-017-2894-z

[2] Visual Paradigm, *What is BPMN?*, [Online]. Loetud aadressil: https://www.visual-paradigm.com/guide/bpmn/what-is-bpmn/ Kasutatud: 21.04.2022.

[3] Kansas State University, *Around the world*, [Online]. Loetud aadressil: https://recservices.k-state.edu/intramurals/rulebooks_handbooks/RBAroundTheWorld.pdf Kasutatud: 21.04.2022.

[4] *Architect Modern Web Applications with ASP.NET Core and Azure*, Updated December 17, 2021. [Online]. Loetud aadressil: https://docs.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/choose-between-traditional-web-and-single-page-apps Kasutatud: 21.04.2022.

[5] freeCodeCamp, *Backend Software Architecture Checklist: How to Build a Product from Scratch*, [Online]. Loetud aadressil: https://www.freecodecamp.org/news/have-an-idea-want-to-build-a-product-from-scratch-heres-a-checklist-of-things-you-should-go-through-in-your-backend-software-architecture/ Kasutatud: 21.04.2022.

[6] JetBrains, *The State of Developer Ecosystem 2021*, [Online]. Loetud aadressil: https://www.jetbrains.com/lp/devecosystem-2021/ Kasutatud: 21.04.2022.

[7] Python Software Foundation, *History and License*, [Online]. Loetud aadressil: https://docs.python.org/3/license.html Kasutatud: 21.04.2022.

[8] GNU, *The JavaScript Trap*, [Online]. Loetud aadressil: https://www.gnu.org/philosophy/javascript-trap.html Kasutatud: 21.04.2022.

[9] Oracle, *Oracle Java SE Licensing FAQ*, [Online]. Loetud aadressil: https://www.oracle.com/java/technologies/javase/jdk-faqs.html#:~:text=You%20can%20download%20the%20latest,from%20jdk.java.net. Kasutatud: 21.04.2022.

[10] Microsoft, *C#*, [Online]. Loetud aadressil: https://dotnet.microsoft.com/en-us/languages/csharp Kasutatud: 21.04.2022.

[11] The PHP Group, *Preface*, [Online]. Loetud aadressil: https://www.php.net/manual/en/preface.php Kasutatud: 21.04.2022.

[12] JetBrains, *Python*, [Online]. Loetud aadressil: https://www.jetbrains.com/lp/devecosystem-2021/python/ Kasutatud: 21.04.2022.

[13] JetBrains, *JavaScript*, [Online]. Loetud aadressil: https://www.jetbrains.com/lp/devecosystem-2021/javascript/ Kasutatud: 21.04.2022.

[14] JetBrains, *Java*, [Online]. Loetud aadressil: https://www.jetbrains.com/lp/devecosystem-2021/java/ Kasutatud: 21.04.2022.

[15] JetBrains, *C#*, [Online]. Loetud aadressil: https://www.jetbrains.com/lp/devecosystem-2021/csharp/ Kasutatud: 21.04.2022.

[16] JetBrains, *PHP*, [Online]. Loetud aadressil: https://www.jetbrains.com/lp/devecosystem-2021/php/ Kasutatud: 21.04.2022.

[17] The Pallets Projects, *License*, [Online]. Loetud aadressil: https://flask.palletsprojects.com/en/2.1.x/license/ Kasutatud: 21.04.2022.

[18] Django Software Foundation, *FAQ: General*, [Online]. Loetud aadressil: https://docs.djangoproject.com/en/4.0/faq/general/#how-is-django-licensed Kasutatud: 21.04.2022.

[19] OpenJS Foundation, *About the OpenJS Foundation*, [Online]. Loetud aadressil: https://openjsf.org/ Kasutatud: 21.04.2022.

[20] IBM, *Java Spring Boot*, [Online]. Loetud aadressil: https://www.ibm.com/cloud/learn/java-spring-boot Kasutatud: 21.04.2022.

[21] Microsoft, *Overview to ASP.NET Core*, [Online]. Loetud aadressil: https://docs.microsoft.com/en-us/aspnet/core/introduction-to-aspnet-core?view=aspnetcore-6.0 Kasutatud: 21.04.2022.

[22] Laravel LLC, *License*, [Online]. Loetud aadressil: https://laravel-guide.readthedocs.io/en/latest/license/ Kasutatud: 21.04.2022.

[23] metanit.com, *Введение в Entity Framework Core*, [Online]. Loetud aadressil: https://metanit.com/sharp/efcore/1.1.php Kasutatud: 21.04.2022.

[24] JetBrains, *Databases*, [Online]. Loetud aadressil: https://www.jetbrains.com/lp/devecosystem-2021/databases/ Kasutatud: 21.04.2022.

[25] SQLite Consortium, *What is SQLite?*, [Online]. Loetud aadressil: https://www.sqlite.org/index.html Kasutatud: 21.04.2022.

[26] *Architect Modern Web Applications with ASP.NET Core and Azure*, Updated December 17, 2021. [Online]. Loetud aadressil: https://docs.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/common-client-side-web-technologies Kasutatud: 21.04.2022.

[27] Facebook, *License*, [Online]. Loetud aadressil: https://github.com/facebook/react/blob/main/LICENSE Kasutatud: 21.04.2022.

[28] Yuxi (Evan) You, *License*, [Online]. Loetud aadressil: https://github.com/vuejs/vue/blob/dev/LICENSE Kasutatud: 21.04.2022.

[29] Google LLC, *License*, [Online]. Loetud aadressil: https://angular.io/license Kasutatud: 21.04.2022.

[30] Red Hat, *What is an IDE?*, [Online]. Loetud aadressil: https://www.redhat.com/en/topics/middleware/what-is-ide Kasutatud: 21.04.2022.

[31] Microsoft, *Visual Studio*, [Online]. Loetud aadressil: https://visualstudio.microsoft.com/free-developer-offers/ Kasutatud: 21.04.2022.

[32] JetBrains, *Rider*, [Online]. Loetud aadressil: https://www.jetbrains.com/rider/buy/#personal Kasutatud: 21.04.2022.

[33] Microsoft, *Visual Studio Code*, [Online]. Loetud aadressil: https://code.visualstudio.com/ Kasutatud: 21.04.2022.

[34] JetBrains, *WebStorm*, [Online]. Loetud aadressil: https://www.jetbrains.com/webstorm/buy/#personal Kasutatud: 21.04.2022.

[35] metanit.com, *Введение в ASP.NET Core*, [Online]. Loetud aadressil: https://metanit.com/sharp/aspnet6/1.1.php Kasutatud: 22.04.2022.

[36] Microsoft, *Architect Modern Web Applications with ASP.NET Core and Azure*, [Online]. Loetud aadressil: https://docs.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/common-web-application-architectures Kasutatud: 22.04.2022.

[37] Microsoft, *Implementing the Repository and Unit of Work Patterns in an ASP.NET MVC Application (9 of 10)*, [Online]. Loetud aadressil: https://docs.microsoft.com/en-us/aspnet/mvc/overview/older-versions/getting-started-with-ef-5-using-mvc-4/implementing-the-repository-and-unit-of-work-patterns-in-an-asp-net-mvc-application Kasutatud: 22.04.2022.

[38] Microsoft, *Develop ASP.NET Core MVC apps*, [Online]. Loetud aadressil: https://docs.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/develop-asp-net-core-mvc-apps Kasutatud: 22.04.2022.

[39] metanit.com, *Аутентификация и авторизация*, [Online]. Loetud aadressil: https://metanit.com/sharp/aspnet6/13.1.php Kasutatud: 22.04.2022.

[40] Evan You, *Introduction*, [Online]. Loetud aadressil: https://vuejs.org/guide/introduction.html#what-is-vue Kasutatud: 22.04.2022.

[41] Evan You, *Ways of Using Vue*, [Online]. Loetud aadressil: https://vuejs.org/guide/extras/ways-of-using-vue.html Kasutatud: 22.04.2022.

[42] Evan You and Eduardo San Martin Morote, *Vue Router*, [Online]. Loetud aadressil: https://router.vuejs.org/introduction.html Kasutatud: 22.04.2022.

[43] Evan You, *Single-File Components*, [Online]. Loetud aadressil: https://vuejs.org/guide/scaling-up/sfc.html Kasutatud: 22.04.2022.

[44] the SASS team, *Documentation*, [Online]. Loetud aadressil: https://sass-lang.com/documentation Kasutatud: 22.04.2022.

[45] Evan You, *What is Vuex?*, [Online]. Loetud aadressil: https://vuex.vuejs.org/ Kasutatud: 22.04.2022.

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

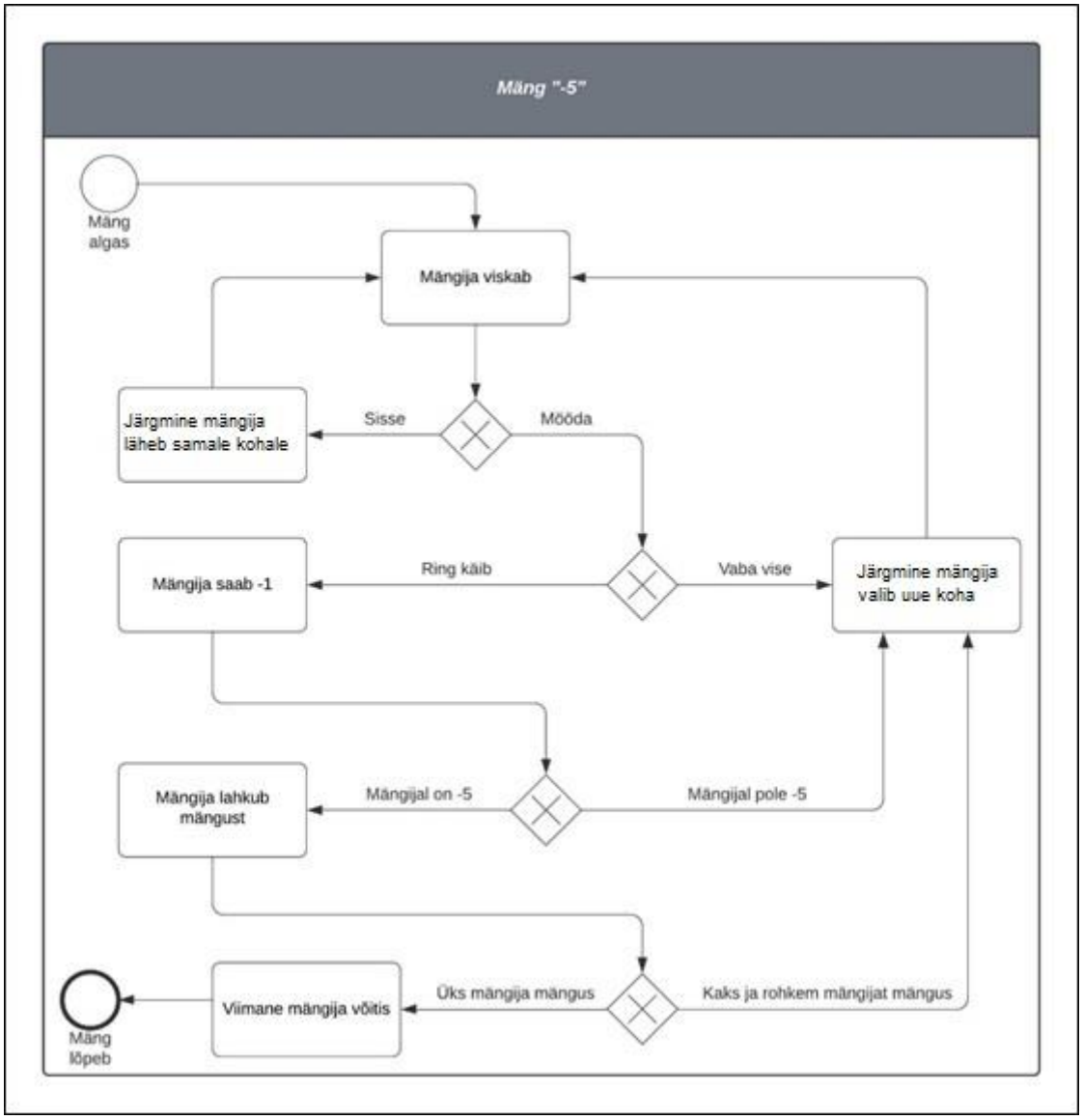
Mina, Aleksei Žavoronkov,

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

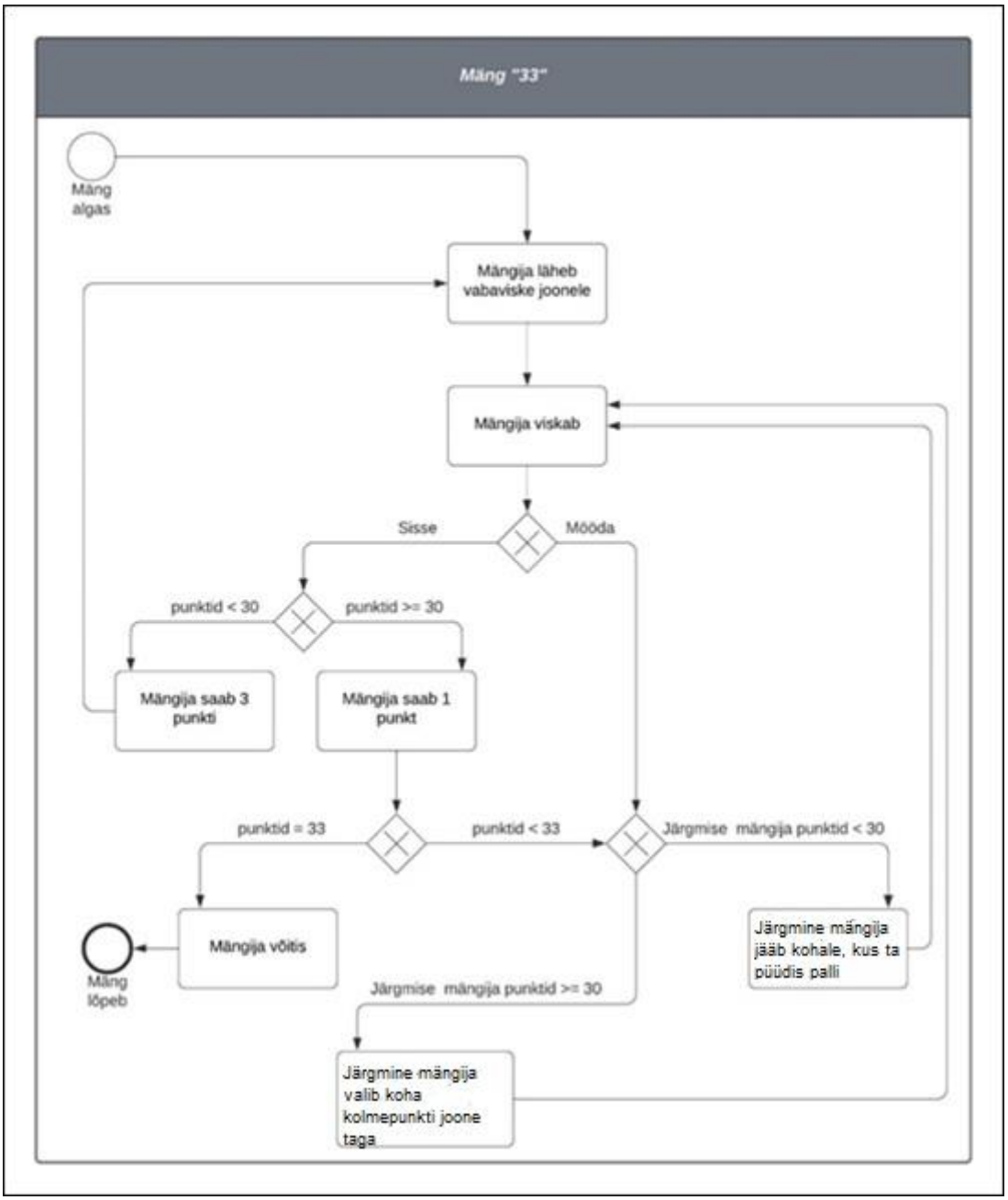
12.05.2022

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

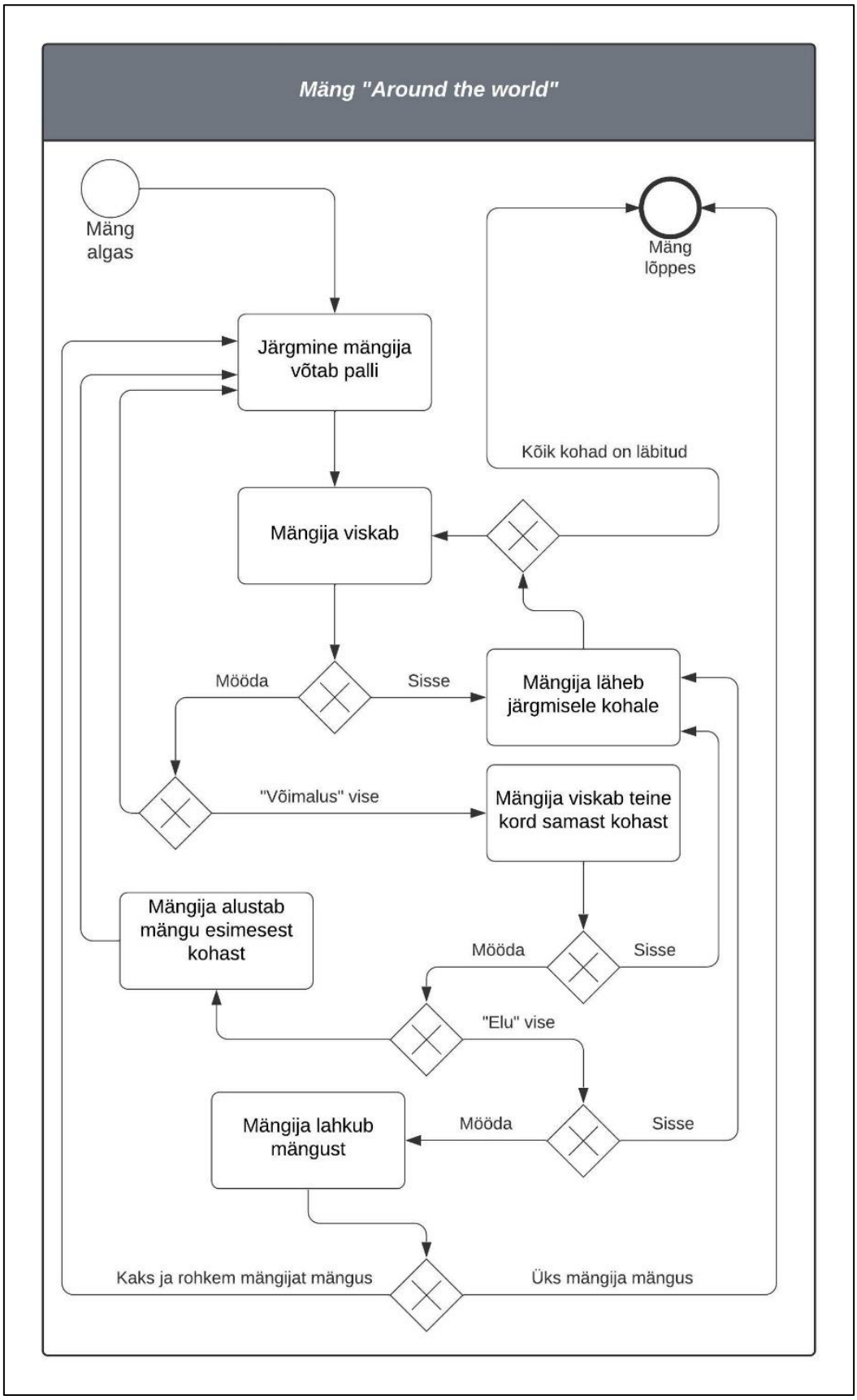
Lisa 2 – Korvpallil baseeruvate mängude mänguprotsesside skeemid



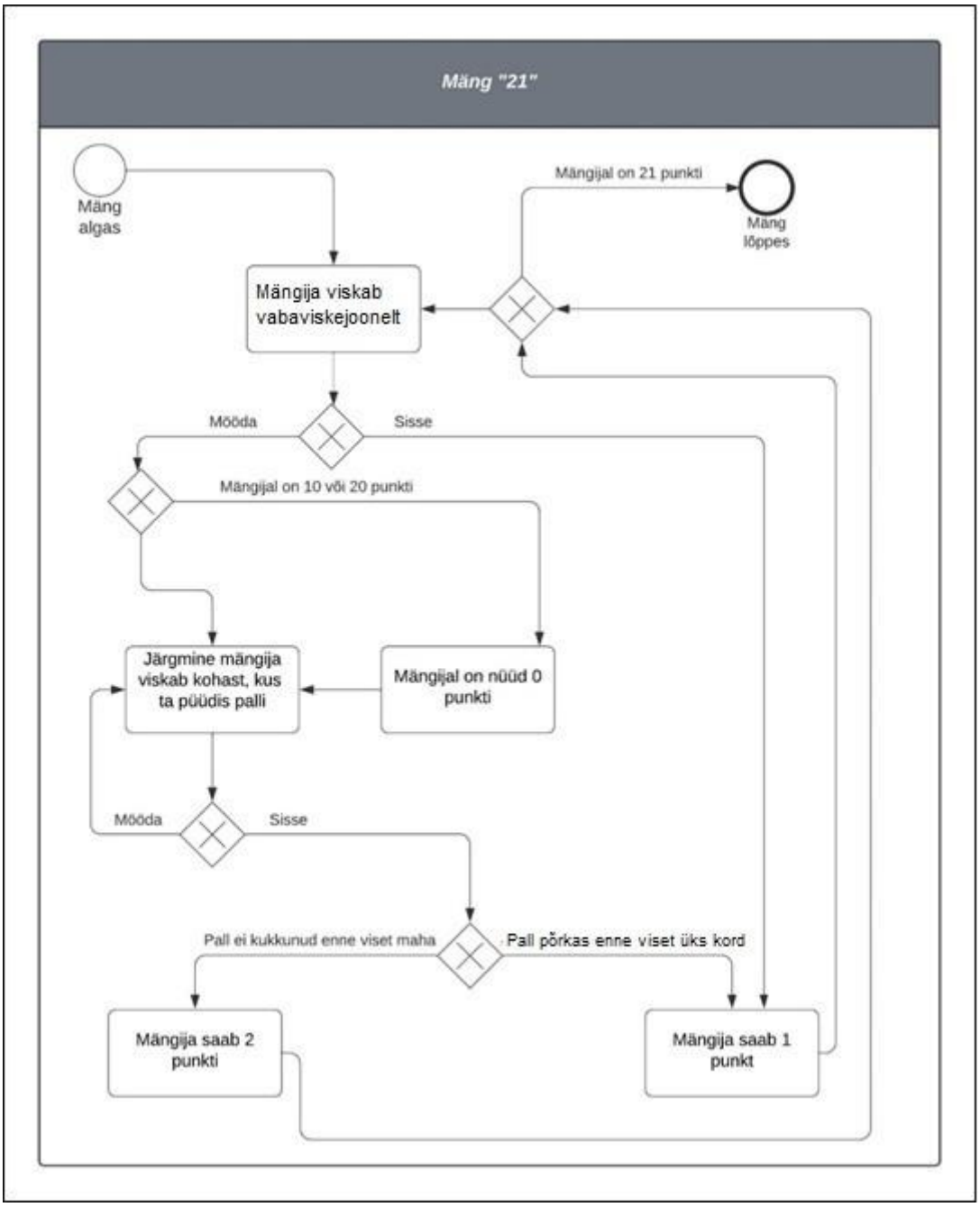
Joonis 1. "-5" mängu skeem.



Joonis 2. "33" mängu skeem.

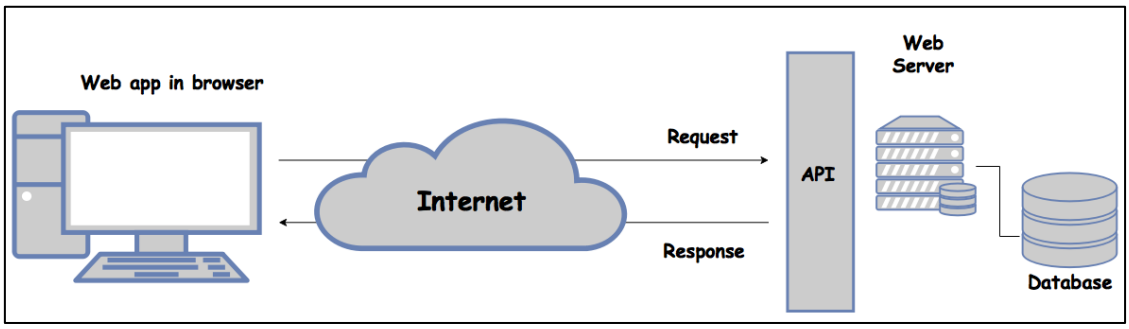


Joonis 3. "Around the world" mängu skeem.



Joonis 4. "21" mängu skeem.

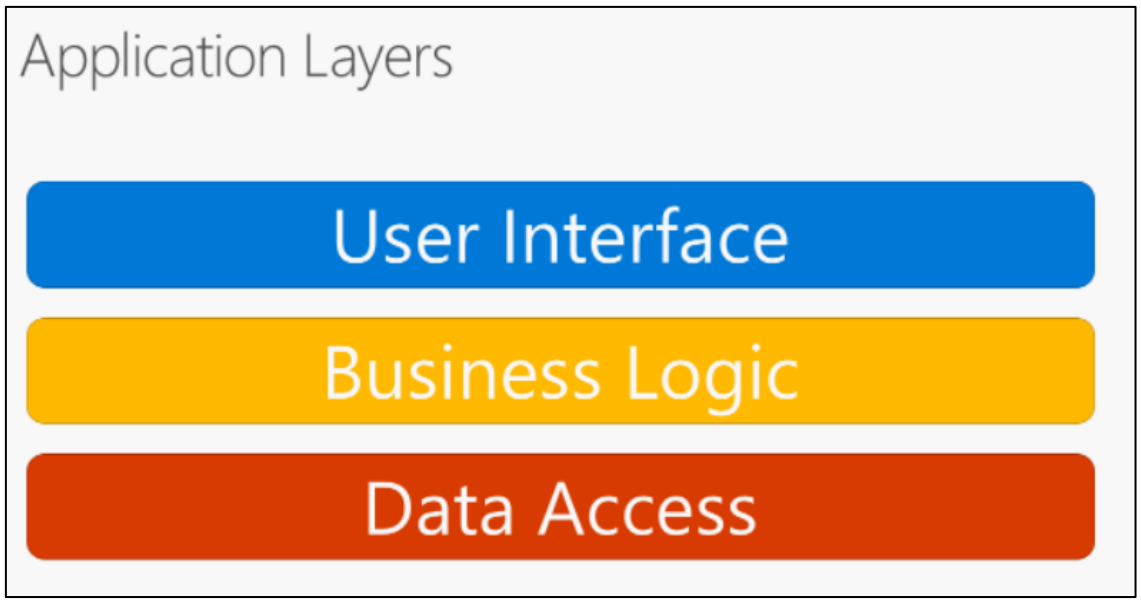
Lisa 3 – Web API arendusmudeli skeem



Joonis 5. Web API arendusmudeli skeem.

G. Martinez, *How to create your own little Restful Web API without getting get lost in the process — Part 2*, [Online]. Võetud aadressilt: https://gabrymartinez.medium.com/how-to-create-your-own-little-restful-web-api-and-not-get-lost-in-the-process-part-2-473400256ce0 Kasutatud: 22.04.2022.

Lisa 4 – Traditsiooniline veebirakenduse arhitektuur



Joonis 6. Traditsiooniline veebirakenduse arhitektuur.

Microsoft, *Common web application architectures*, [Online]. Võetud aadressilt: https://docs.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/common-web-application-architectures Kasutatud: 22.04.2022.

Lisa 5 – Repository arendusmustri liides

```
public interface IRepository<TKey, TEntity, TDto>
      where TKey : IEquatable<TKey>
      where TEntity : class
      where TDto : class
{
      Task<IEnumerable<TDto>> GetAllAsync();
      Task<TDto?> GetOneAsync(TKey id);
      void Add(TDto dto);
      void Edit(TDto dto);
      void Delete(TDto dto);
}
```

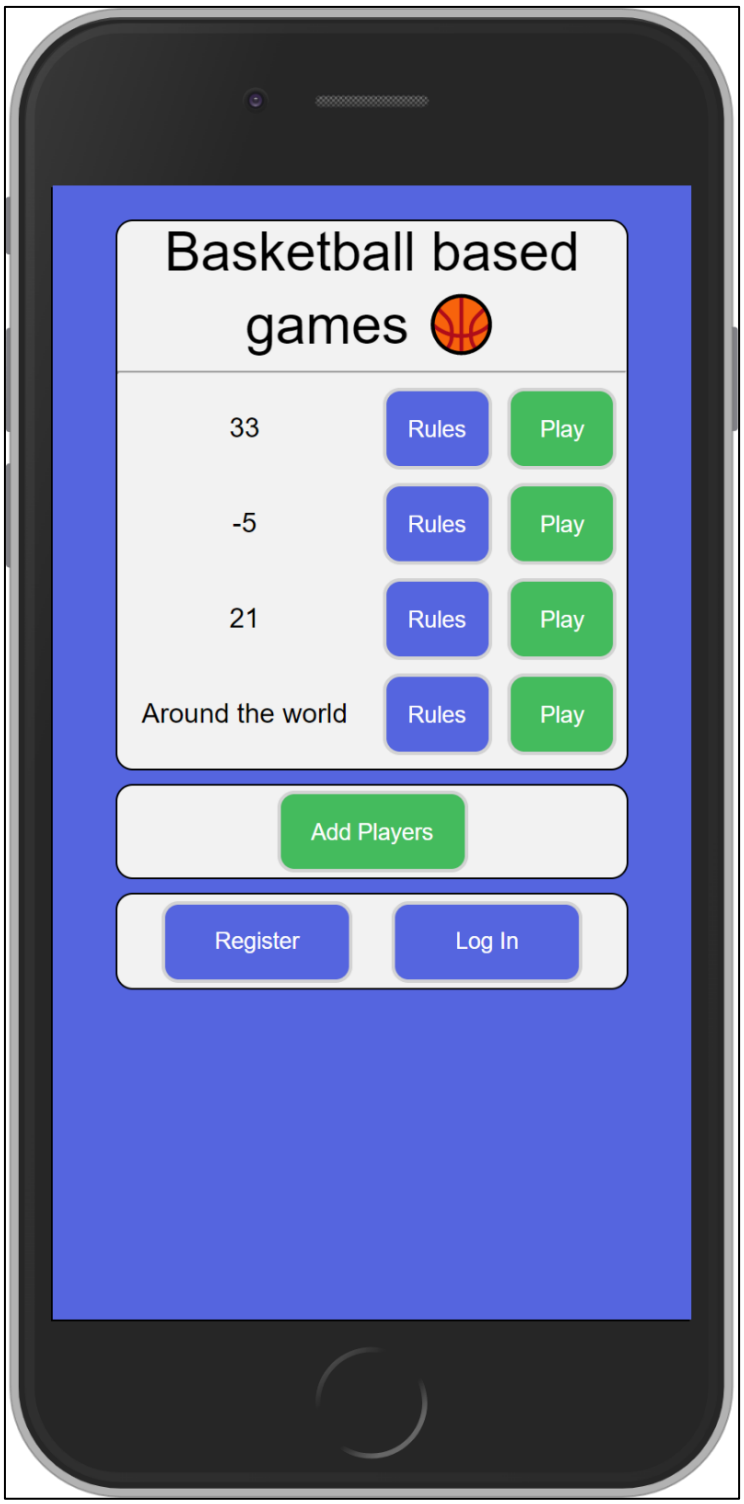
Joonis 7. Repository arendusmustri liides.

Lisa 6 – Unit of Work arendusmustri liides

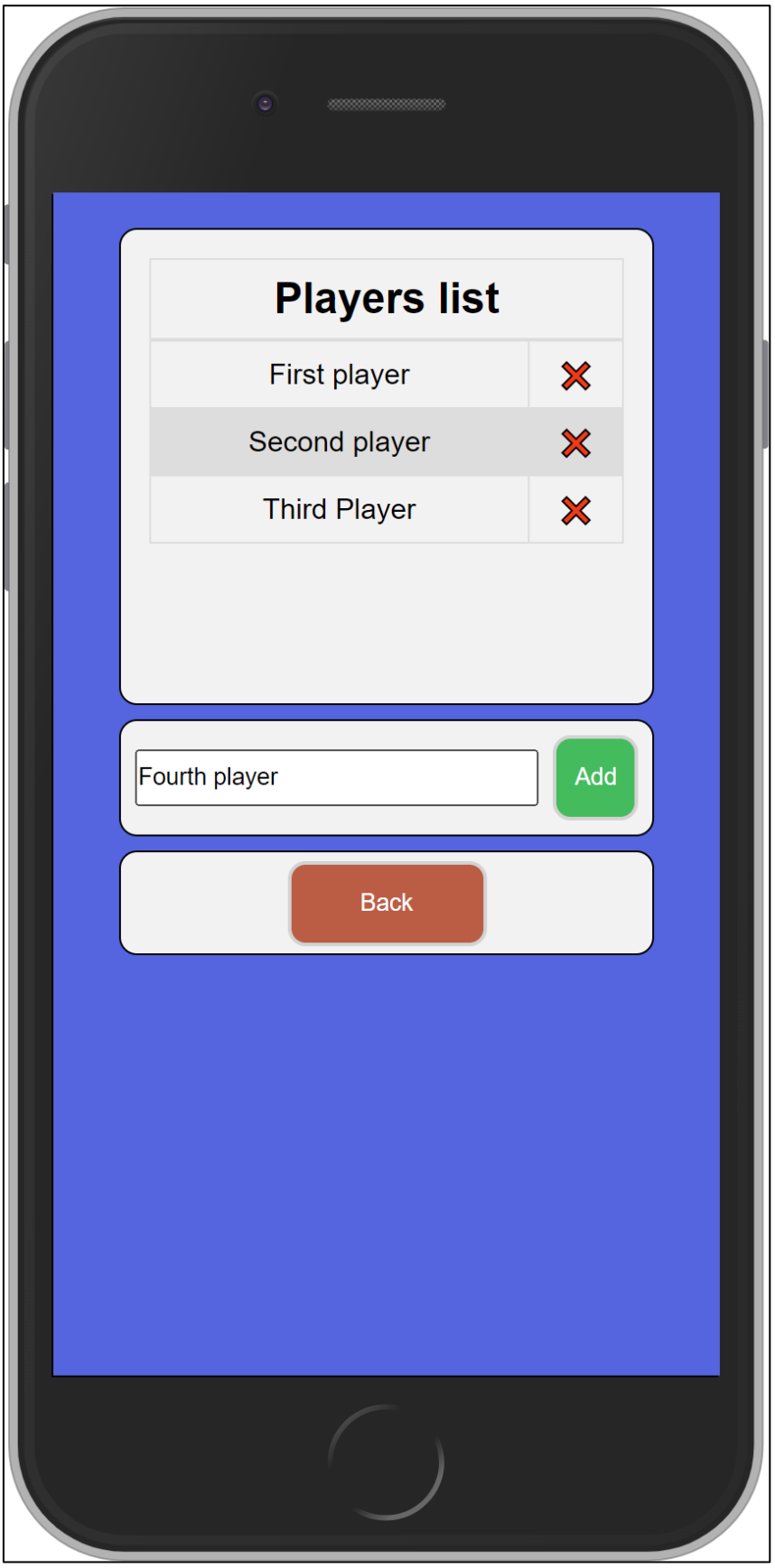
```
public interface IUnitOfWork
{
      IGamesRepository Games { get; }
      IUsersRepository Users { get; }
      IFriendshipsRepository Friendships { get; }
      IPlayedGamesRepository PlayedGames { get; }
      Task<int> SaveChangesAsync();
}
```

Joonis 8. Unit of Work arendusmustri liides.

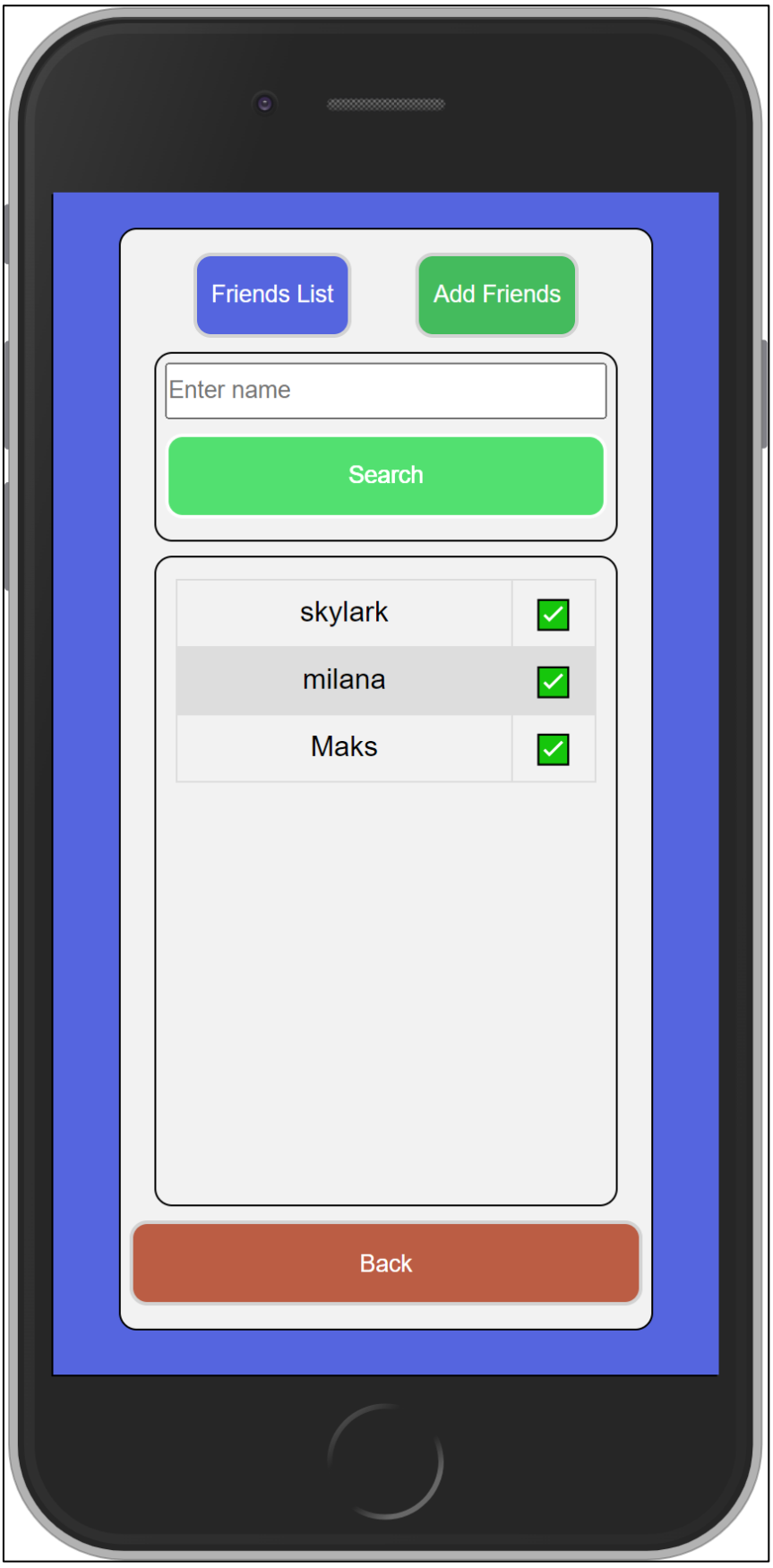
Lisa 7 – Veebirakenduse vaated



Joonis 9. Mängude loetelu vaade.

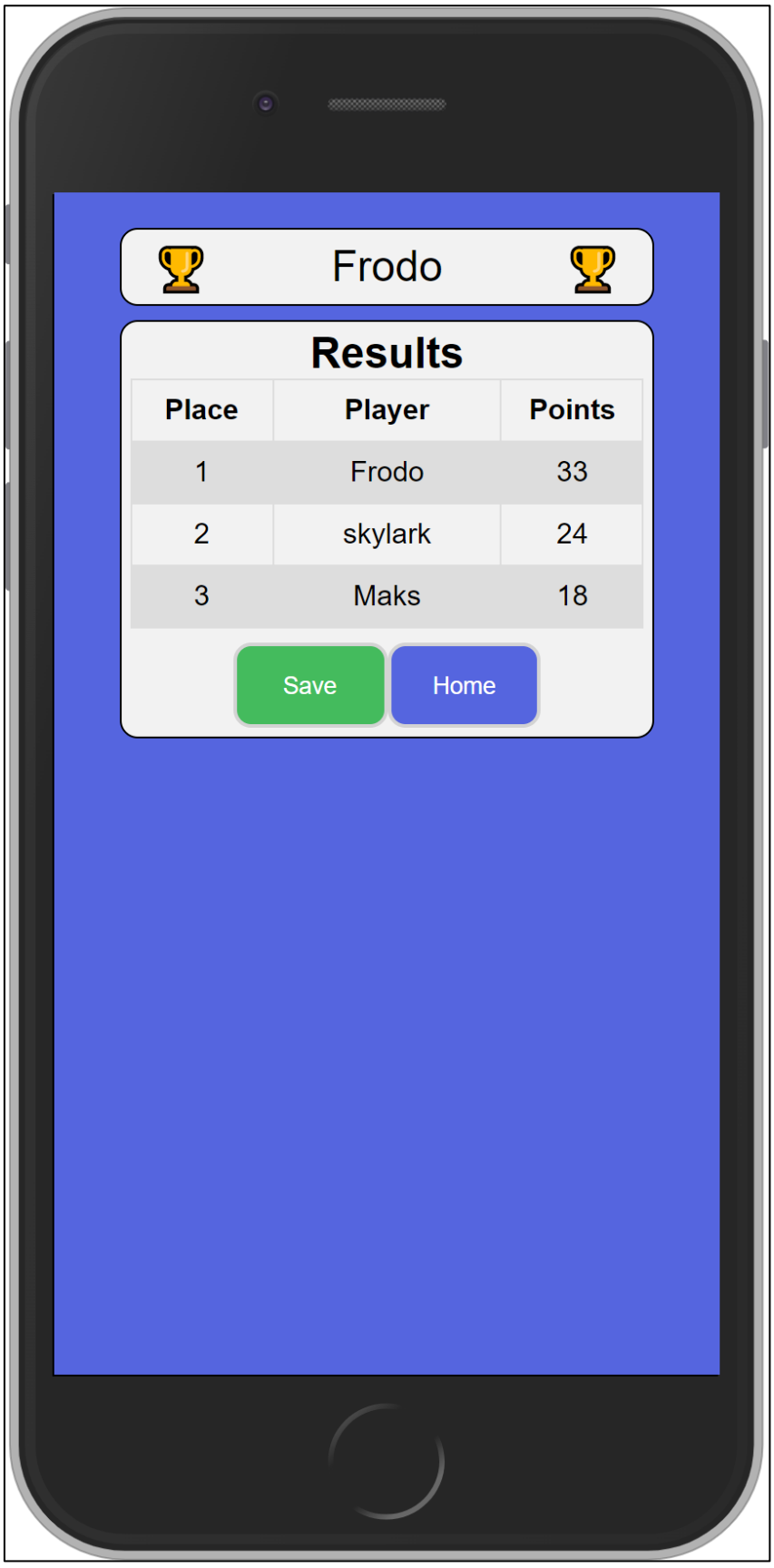


Joonis 10. Mängijate lisamise vaade.



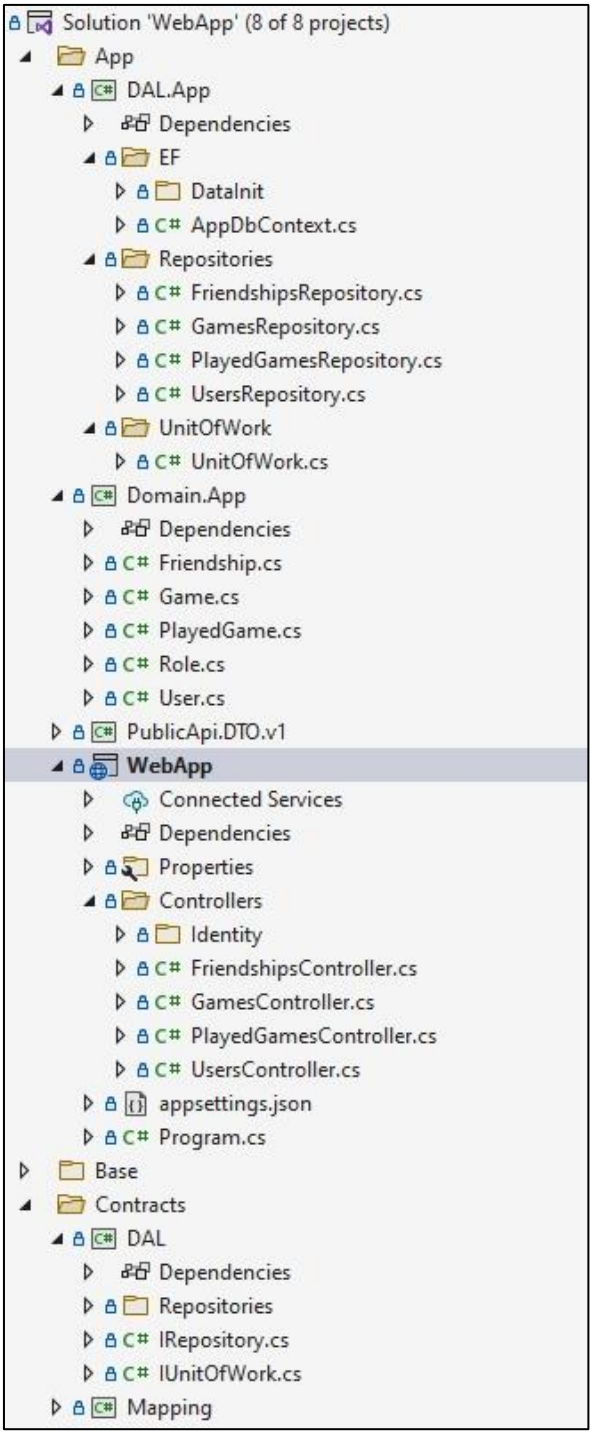
Joonis 11. Sõprade otsimise vaade.

Joonis 12. "Around the world" mängu vaade.



Joonis 13. Lõpptulemuste vaade.

Lisa 8 – Teenusepoolse keskkonna projekti struktuur



Joonis 14. Teenusepoolse keskkonna projekti struktuur.

Lisa 9 – Kliendipoolse keskkonna projekti struktuur

Joonis 15. Kliendipoolse keskkonna projekti struktuur.

Lisa 10 – Veebirakenduse versioonihaldused

Teenusepoolne keskkond: https://github.com/alzavo/basketball-based-games-backend

Kliendipoolne keskkond: https://github.com/alzavo/basketball-based-games-client