TANSZÉKVEZETŐ

SZAKDOLGOZAT FELADAT

Fazekas Bence
Mérnökinformatikus hallgató részére

Android alapú testedzést támogató alkalmazás fejlesztése

A mai világban egyre fontosabb az emberek számára, hogy egészségesen táplálkozzanak és rendszeresen sportoljanak. Észrevehetően, a fiatalok többsége a sportolást edzőtermekben végzik, amivel semmi baj nincs, de sokan nincsenek tisztában vele, hogy hogyan is kell a különböző gyakorlatokat szabályosan végezni. Ez számos veszélyt magában foglal. Fizikai sérüléseket, esetleg sérvek kialakulását, valamit izületi sérüléseket, amik nem feltétlen azonnal, hanem az idő múlásával jelentkeznek. A regenerálódási idejük hosszú lefolyású, legtöbbször orvosi beavatkozás, műtétek szükségesek a teljes felépüléshez.

A dolgozat célja egy olyan android alkalmazás fejlesztése, amely egy adatbázisban tárolt adatokon végzett számítások alapján napi kalóriabevitelt tanácsol a felhasználók számára. Továbbá megtekinthetik, hogy hogyan kell a különböző izomcsoportokra vonatkozó gyakorlatokat szabályosan elvégezni, valamint saját edzéstervet állíthatnak össze.

A hallgató feladatának a következőkre kell kiterjednie:

- Mutassa be a Backend architektúra felépítését, az egyes komponensek feladatkörét!
- Mutassa be az elkészített MySQL adatbázist!
- Készítse el az Android alkalmazás komponenseit!
- Mutassa be az alkalmazás tesztelési szintjeit, a tesztelés megvalósítását!

Tanszéki konzulens: Dr. Ekler Péter

Budapest, 2018. szeptember 26.

Dr. Charaf Hassan
egyetemi tanár
tanszékvezető

Budapest Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Automatizálási és Alkalmazott Informatikai Tanszék
H-1117 Budapest, Magyar tudósok krt 2. Q.B.207
Telefon: 463-2870. Fax: 463-2871
www.aut.bme.hu

Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Automatizálási és Alkalmazott Informatikai Tanszék

Fazekas Bence

ANDROID ALAPÚ TESTEDZÉST TÁMOGATÓ ALKALMAZÁS FEJLESZTÉSE

KONZULENS

DR. EKLER PÉTER

BUDAPEST, 2018

Tartalomjegyzék

Összefoglaló **5**

Abstract **6**

1 Bevezetés **7**

1.1 Témaválasztás 7

1.2 Hasonló megoldások 8

1.3 A szakdolgozat felépítése 9

2 Alkalmazás ismertetése **10**

2.1 Feladatspecifikáció 10

2.2 Felhasználók 10

2.2.1 Felhasználó 11

2.2.2 Nem regisztrált felhasználó 11

2.3 Tervezett funkciók 12

2.3.1 Regisztráció 12

2.3.2 Bejelentkezés 14

2.3.3 Gyakorlatok megjelenítése 16

2.4 Kalóriaigény számítás 17

3 Felhasznált technológiák **18**

3.1 Android platform 18

3.2 Node.js 19

3.3 Express 19

3.4 MySQL 19

3.4.1 Lekérdezések 20

3.4.2 Adatok bevitele 20

3.4.3 Adatok módosítása 20

3.4.4 Adatok törlése 20

3.5 JSON Web Token 21

3.6 REST 21

4 Architektúra bemutatása **23**

4.1 Háromrétegű architektúra 23

5 Adatbázis **25**

5.1 Kapcsolatok a táblák között 25

5.2 Users tábla 25

5.3 Trainings tábla 26
5.4 UsersTrainings tábla 27
6 Szerveralkalmazás 28
6.1 Felépítés 28
6.2 Controllers 28
6.2.1 UserController 29
6.2.2 TrainingController 30
6.2.3 UsersTrainingsController 30
6.3 Counters 31
6.4 Database 31
6.5 Gifs 31
6.6 Middleware 31
6.7 Validators 32
7 Mobilalkalmazás 33
7.1 Felépítés 33
7.2 Activitys 33
7.3 Adapters 34
7.4 Application 34
7.5 Fragments 34
7.6 UserInformation 35
7.7 RecyclerViewElements 36
7.8 Kérés a Szerveralkalmazás felé 36
8 Mobilalkalmazás használatának a bemutatása 37
9 Tesztelés 40
9.1 Mobilalkalmazás tesztelése 40
9.2 Rest API tesztelése 41
10 Összegzés és továbbfejlesztési lehetőségek 42
10.1 Fejlesztési folyamat összefoglalása 42
10.2 Továbbfejlesztési lehetőségek 42
11 Irodalomjegyzék 43

HALLGATÓI NYILATKOZAT

Alulírott **Fazekas Bence**, szigorló hallgató kijelentem, hogy ezt a szakdolgozatot meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző, cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy hitelesített felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Kelt: Budapest, 2018. 12. 07

...…….

Fazekas Bence

Összefoglaló

A mai világban egyre fontosabb az emberek számára, hogy egészségesen táplálkozzanak és rendszeresen sportoljanak. Észrevehetően a fiatalok többsége a sportolást edzőtermekben valósítja meg, amivel semmi baj nem lenne, de sokan nincsenek tisztában vele, hogy hogyan is kell különböző gyakorlatokat szabályosan elvégezni. Ez számos veszélyt magában foglal. Fizikai sérüléseket, esetleg sérvek, valamint ízületi sérülések kialakulását, amik nem feltétlen azonnal, hanem az idő múlásával jelentkeznek. A regenerálódási idejük hosszú lefolyású, legtöbbször orvosi beavatkozás, műtétek szükségesek a teljes felépüléshez.

Ezért szakdolgozatom célja egy olyan Android alkalmazás fejlesztése, amely segítségével a felhasználók meg tudják tekinteni különböző izomcsoportokhoz tartozó gyakorlatokat, hogyan is kell szabályosan elvégezni. Továbbá saját edzéstervet tudnak maguknak létrehozni, valamint tisztában lesznek vele, hogy napi szinten, mennyi az ajánlott kalóriabevitel számukra.

A szakdolgozat ennek az Android alkalmazásnak a fejlesztését írja le. Az elkészült program lehetővé teszi felhasználók regisztrálását, bejelentkezését. Általam elkészített GIF-ek alapján lehet megtekinteni, hogyan kell szabályosan elvégezni a gyakorlatokat és ezeket a gyakorlatokat lehet hozzáadni saját edzéstervhez. A Node.js-ben írt szerver program a felhasználói adatokat és a gyakorlatokat egy adatbázisba küldi fel és ott kerülnek elmentésre.

Abstract

Nowadays, it is getting more and more important for people, to eat healthy and do some sports regularly. Noticeably the most of the youth choose the gym for doing exercises, which would not be a problem, but a lot of people do not know how to make various exercises properly. This contains several dangers. Physical injuries, maybe development of joint diseases and hernia, which may not appear immediately only after a long time. The regeneration takes a considerable period of time, in the most cases medical interventions, operations are needed to the complete recovery.

That is why the aim of my examination paper is to develop an Android application which helps the users to view the exercises to the different muscle groups to be aware of how to make them properly. Furthermore, they can create their own training plan, moreover they will know how much is the perfect amount of calorie intake on a daily basis.

The examation paper describes the development of this Android application. The complete program allows the registration and login of the users. It can be viewed by gifs made by me how to do the exercises regularly and these exercises can be added to the own training plan. The program on the server, written in the Node.js sends the user data and exercises to a database stored there.

1 Bevezetés

A bevezetésben bemutatásra kerül a témaválasztásom indoka, hasonló alkalmazások, továbbá a szakdolgozat felépítése.

1.1 Témaválasztás

Témaválasztásom során fontos szempont volt, hogy egy olyan témát találjak, ami hozzám közel áll és szívesen foglalkozok vele. Lassan második éve, hogy majdnem napi szinten konditerembe járok, ezért jött ez az ötlet, hogy egy ehhez kapcsolódó feladatot válasszak. Amikor elkezdtem edzőterembe járni én se voltam még tisztában nagyon sok dologgal, de vettem a fáradtságot és sok mindennek utána olvastam. Sokan nem szeretnek, vagy nem mernek kérdezni a tapasztaltabbaktól, mert kínosnak érzik, de ezzel kockára teszik az egészségüket, valamint fejlődni se fognak rendesen.

Nagy könnyítés lett volna számomra, ha lett volna egy ilyen alkalmazás telefonra. Napjainkban mobilkészülékkel szinte minden ember rendelkezik már, az életben szinte mindenhol segítségünkre lehet a telefonunk. Az utóbbi egy évben döntöttem el, hogy telefonra való fejlesztéssel szeretnék foglalkozni a jövőben. Egyre több program jelenik meg ezekre az eszközökre, valamint egyszerű kezelhetőség és könnyű hordozhatóság miatt ez a szám rohamosan nő, és nőni is fog.

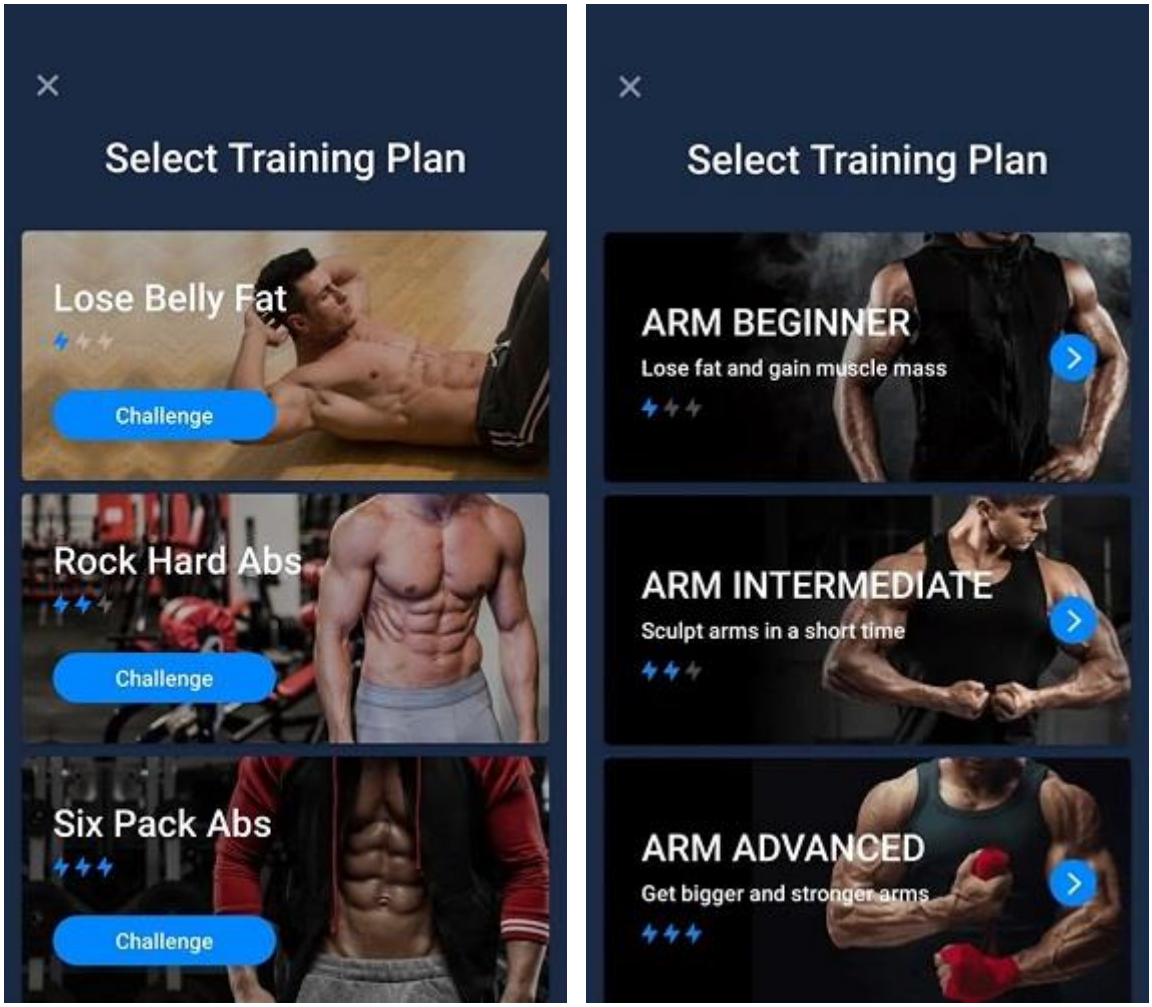
1.1. ábra: Android és iOS logója

Mobileszközök gyors terjedésének köszönhetően, szoftverekre való igény is egyre inkább növekszik. Jelenleg két nagy mobil operációs rendszer van a piacon, ezek az iOS, valamint az Android.

1.2 Hasonló megoldások

Mielőtt neki álltam volna fejleszteni a *BuildBody* nevű alkalmazásomat, utána jártam, hogy milyen hasonló mobilalkalmazások érhetőek el a felhasználók számára. A Play Áruházban már találhatóak hasonló alkalmazások, de szinte kivétel nélkül hiányérzetem volt, amikor kipróbáltam ezeket az alkalmazásokat.

A legnagyobb probléma az volt, hogy nem volt egy egységes alkalmazás, ami minden testrészhez tartalmazta volna a gyakorlatokat. Minden alkalmazás külön – külön egy testrészre volt szabva, vagyis ha valaki minden testrészhez szeretne segítséget igénybe venni, akkor legalább 8 alkalmazást kell feltennie a telefonjára alsó hangon.



1.2 ábra: Hasonló alkalmazások a Play Áruházból [1, 2]

Másik probléma, amivel találkoztam az az, hogy ezek kész edzésterveket tartalmaznak, és nem lehet saját magunknak edzéstervet összeállítani a meglévő gyakorlatokból. Az ötletet, hogy vannak külön kezdőknek, haladóknak, és tapasztaltaknak is gyakorlat kifejezetten jónak tartom, de amikor megnéztem a gyakorlatokat igencsak csalódott voltam. Nem igazán tükrözte a különbséget a kategóriák között. Legtöbb helyen csak az ismétlésszámban tértek el egymástól, valamint nem tartalmaztak súlyzókkal való edzéshez tanácsokat, csak szabad súllyal történő gyakorlatokat mutattak be.

1.3 A szakdolgozat felépítése

A következő fejezetekben ismertetni fogom az elkészült alkalmazást, valamint az elkészítési folyamatot. A felépítés a következőképpen alakul:

- A második fejezetben ismertetni fogom az elkészítendő alkalmazást, valamint annak funkcióit, és a felhasználókat.
- A harmadik fejezetben az elkészítéshez felhasznált technológiákra fogok kitérni.
- A negyedik fejezetben az architektúra bemutatásáról lesz szó.
- Az ötödik fejezetben az elkészült Adatbázis kerül bemutatásra.
- A hatodik fejezetben az elkészült Szerveralkalmazás kerül bemutatásra.
- A hetedik fejezetben az elkészült Android alkalmazás kerül bemutatásra.
- A nyolcadik fejezetben a mobilalkalmazás használatát fogom bemutatni.
- A kilencedik fejezetben a tesztelésről lesz szó.
- Végül a tizedik fejezetben a munkám összegzéséről és továbbfejlesztési lehetőségekről lesz szó.

2 Alkalmazás ismertetése

Ebben a fejezetben, a szakdolgozat alatt elkészített feladatnak, a részletes specifikációját fogom ismertetni. Bemutatásra kerülnek továbbá, a felhasználók és az alkalmazástól elvárt funkciók is, valamint a napi kalóriaszükségletnek a számítási módját is ismertetni fogom.

2.1 Feladatspecifikáció

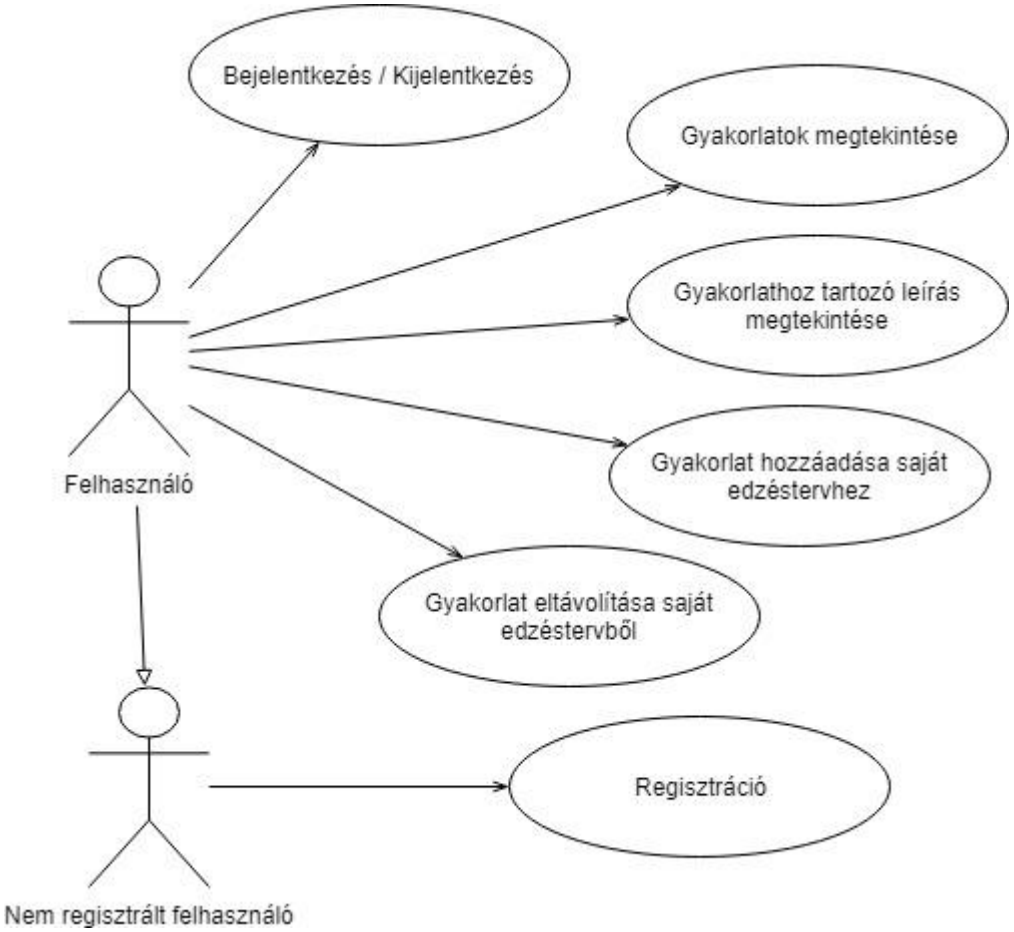
A feladat egy Android alkalmazás elkészítése, amelyben a felhasználók meg tudják tekinteni, hogy hogyan is tudják szabályosan elvégezni, különböző testrészekhez tartozó gyakorlatokat. Ezeket a gyakorlatokat a felhasználók hozzá tudják adni saját edzéstervükhöz, ahol már csak az általuk kiválasztott gyakorlatok jelennek meg a különböző testrészekhez.

Használat előtt a felhasználóknak regisztrálniuk kell a mobilalkalmazásban, ezt követően pedig be kell jelentkezniük. A felhasználók által megadott adatokat, a gyakorlatokat, valamint a felhasználók által kiválasztott gyakorlatok MySQL adatbázisban kerültek eltárolásra. A gyakorlatok bemutatása GIF-ek formájában valósul meg, valamint rövid leírást is olvashatnak, hogy hogyan is kell szabályosan kivitelezni a gyakorlatot. A szükséges GIF fájlok tárolása a szerver egy dedikált könyvtárában valósul meg.

2.2 Felhasználók

Az elkészült alkalmazásban minden felhasználó ugyanolyan hatáskörrel rendelkezik, mint mások. Nem készült külön felhasználói csoport olyan felhasználóknak, akik esetleg már tapasztaltabbak az edzések terén, és esetleg szeretnének ők is, hasznos tanácsokat nyújtani az újoncok számára. Az alkalmazásban ennek ellenére, mégis két felhasználói csoport különíthető el:

- nem regisztrált felhasználó,
- felhasználó.



2.1. ábra: Use case diagram

2.2.1 Felhasználó

A felhasználónak tekintjük azt a személyt, aki már regisztrált az alkalmazásba. Számára lehetőség van az alkalmazás minden funkcióját kihasználni.

- bejelentkezés / kijelentkezés,
- gyakorlatok megtekintése,
- gyakorlatokhoz tartozó leírás megtekintése,
- gyakorlat hozzáadása saját edzéstervhez,
- gyakorlat eltávolítása saját edzéstervből.

2.2.2 Nem regisztrált felhasználó

A nem regisztrált felhasználó kizárólag a regisztráció folyamatát tudja elindítani, máshoz nem fér hozzá. Az adatok kitöltése, és a regisztráció gomb megnyomása után az alábbi lehetőségek következhetnek be:

- Sikeres regisztráció, visszakerülünk a bejelentkező felületre.
- Sikertelen regisztráció:
    - A felhasználó már létezik.
    - Valamelyik adat, esetleg adatok nem megfelelően lettek kitöltve.

A felhasználó létezésének tényleges jelentése az, hogy ez az email cím már szerepel az adatbázisunkban, vagyis már regisztráltak vele.

2.3 Tervezett funkciók

A legfontosabb funkciók, amelyek a felhasználók és a nem regisztrált felhasználók tudnak használni:

- regisztráció,
- bejelentkezés / kijelentkezés,
- gyakorlatok megtekintése,
- gyakorlathoz tartozó leírás megtekintése,
- gyakorlatok hozzáadása saját edzéstervhez,
- gyakorlat eltávolítása saját edzéstervből.

Néhány bonyolultabb funkcióhoz készítettem folyamatdiagramot is, a könnyebb érthetőség kedvéért. A különböző diagramokat egy online tool [3] segítségével készítettem el.

2.3.1 Regisztráció

A 2.3-as ábrán a regisztráció folyamatát tekinthetjük meg. A felhasználó letöltötte és sikeresen telepítette a mobilalkalmazást. Az alkalmazás indítása során a felhasználónk még a nem regisztrált felhasználók közé tartozik, vagyis csak a regisztráció funkciót érheti el. Regisztráláskor számos adatot kell megadnunk, amelyek a következők:

- név,
- email,
- jelszó,

- nem,
- aktivitási szint,
- életkor,
- súly,
- magasság.

A felhasználó által megadott adatokat, amiket képes, a mobil alkalmazás fogja verifikálni. Lehetséges hibaesetek lehetnek a következők:

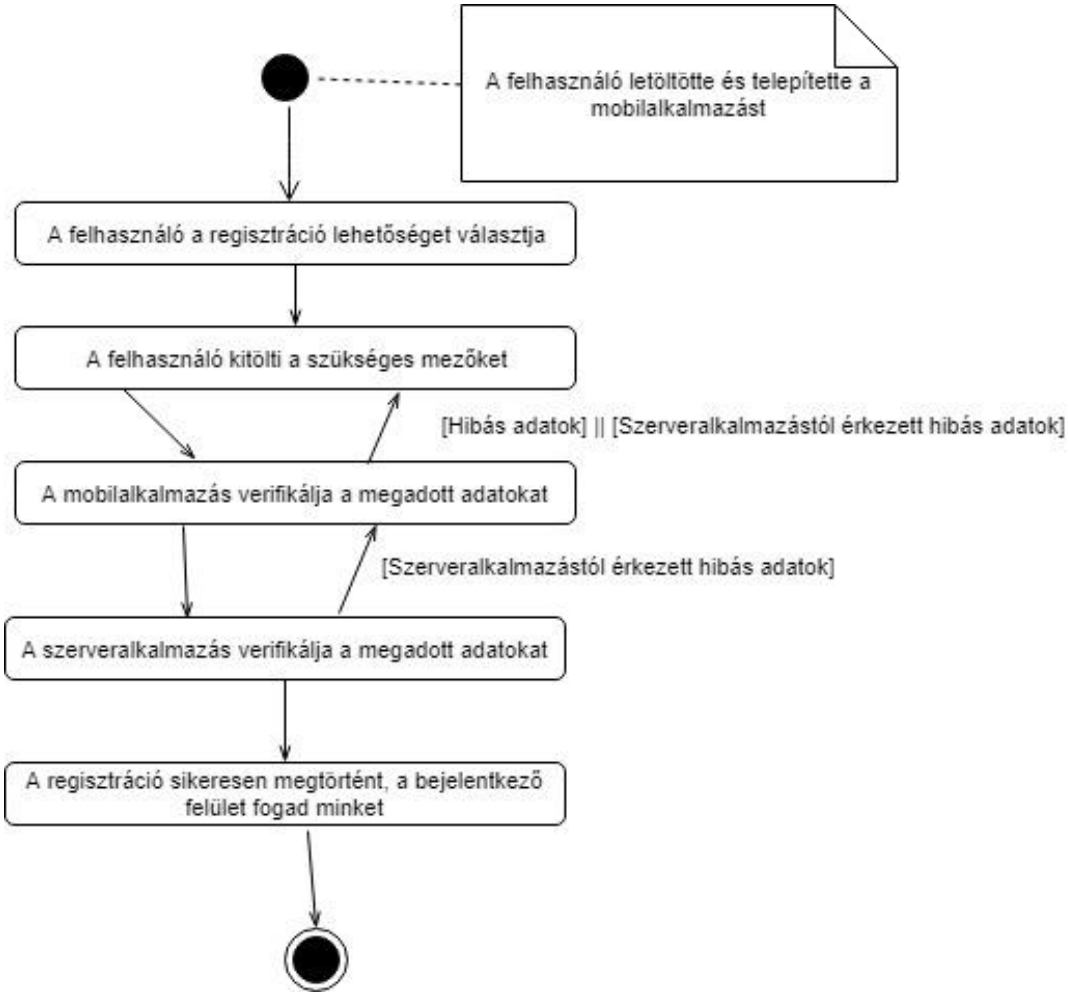
- Nem töltötte ki az adott mezőt.
- Nem megfelelő email formátum.
- Nem egész számban adta meg az adott mező értékét.

A nem egész számban adta meg az adott mező értékét hibaüzenet akkor jelentkezik, amikor az életkor, súly, vagy magasságot töltjük ki, ugyanis itt egész számot vár el az alkalmazás. Ez azért van, mert a MySQL adatbázisban létrehozott táblában, ezeket a mezőket Integerként definiáltam, azaz egész számként várja el az adatokat. Az Aktivitási szint mező egy kis magyarázatra szorul, hogy mit is jelent valójában. Három lehetőség közül választhatunk:

- könnyű (Napi 2 óra mozgásnál kevesebbet végzünk),
- közepes (Napi 2-4 óra mozgást végzünk),
- nehéz (Napi 4 óránál több mozgást végzünk).

A Nem, Aktivitási szint, Életkor, Súly, Magasság megadására azért van szükség, mert ezekkel a paraméterekkel számítódik ki, a napi ajánlott kalóriamennyiség bevitele. A Szerveralkalmazás a magadott adatokon olyan verifikálásokat végez, amelyek mobilalkalmazáson keresztül nem lehetségesek. Ide tartozik például:

- A megadott email cím már foglalt.



2.2. ábra: A regisztráció folyamat Activity diagramja

2.3.2 Bejelentkezés

A 2.4-es ábrán a bejelentkezés folyamatát tekinthetjük meg. A felhasználó sikeres regisztrációt követően megpróbál bejelentkezni az alkalmazásba. A mobilalkalmazás és a Szerveralkalmazás közösen eldöntik az elején, hogy rendelkezik-e a felhasználó érvényes tokennel. Két eset lehetséges:

- A felhasználó tokenje lejárt.
- A felhasználó érvényes tokennel rendelkezik.

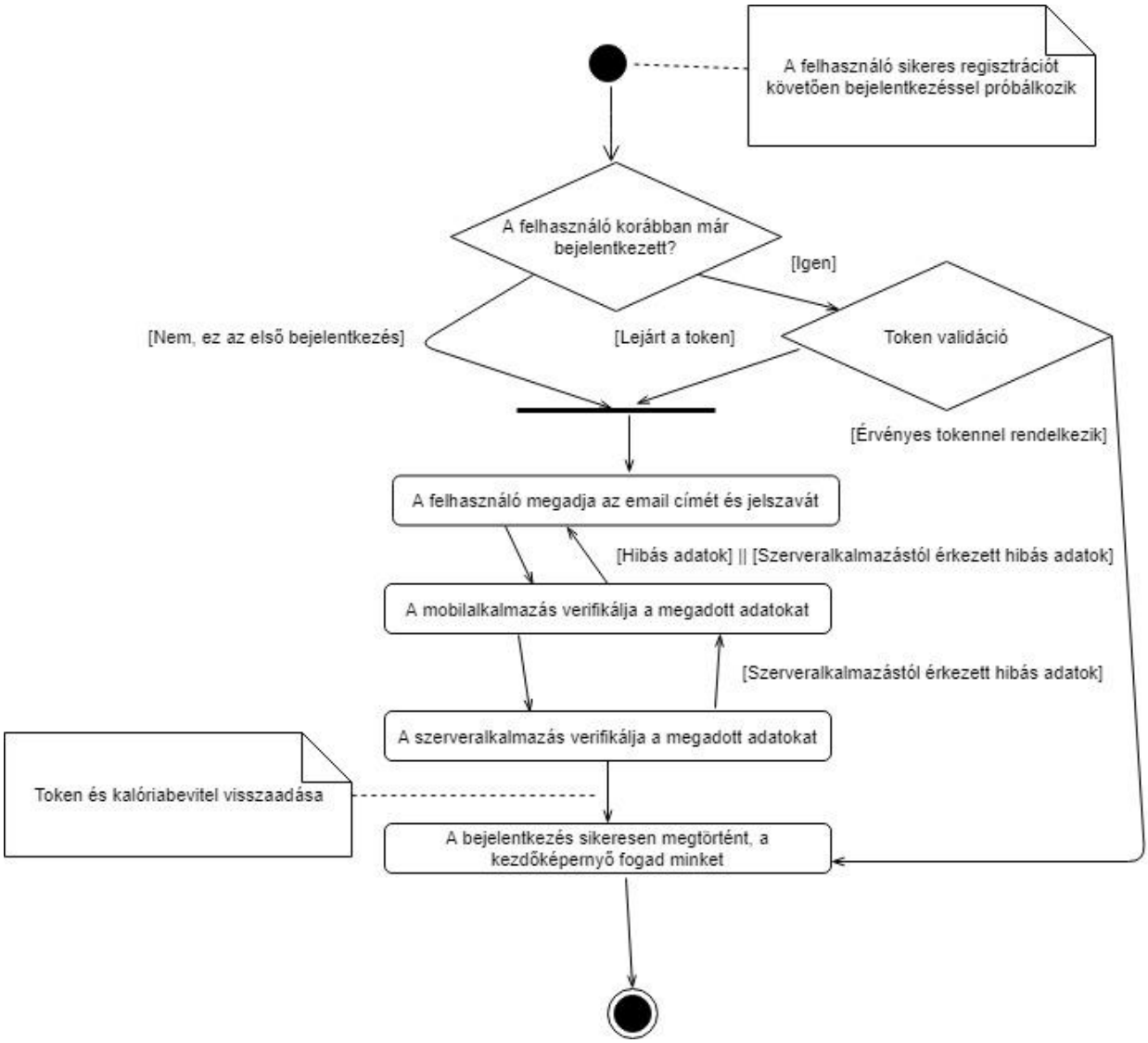
Amennyiben lejárt a felhasználó tokenje, akkor olyan, mint ha most először szeretne bejelentkezni, vagyis a bejelentkező felület fogja őt fogadni. Itt a felhasználó email cím és hozzá tartozó jelszó párossal fogja azonosítani magát. A megadott adatokat, amiket képes, a mobil alkalmazás fogja verifikálni. Lehetséges hibaesetek lehetnek a következők:

- Nem töltötte ki az email cím mezőt.
- Nem töltötte ki a jelszó mezőt.
- Nem megfelelő email formátum.

A Szerveralkalmazás a magadott adatokon olyan verifikálásokat végez, amelyek mobilalkalmazáson keresztül nem lehetségesek. Ide tartoznak például:

- Nincs ilyen email cím.
- Nem jó az email – jelszó páros.

Ha a felhasználó érvényes tokennel rendelkezik, valamint ha sikeresen megadta az email cím – jelszó párost, akkor a kezdőképernyő fogad minket. Ebben az esetben a szervertől válaszként a token és a napi szinten szükséges kalóriabevitelt kapjuk vissza.



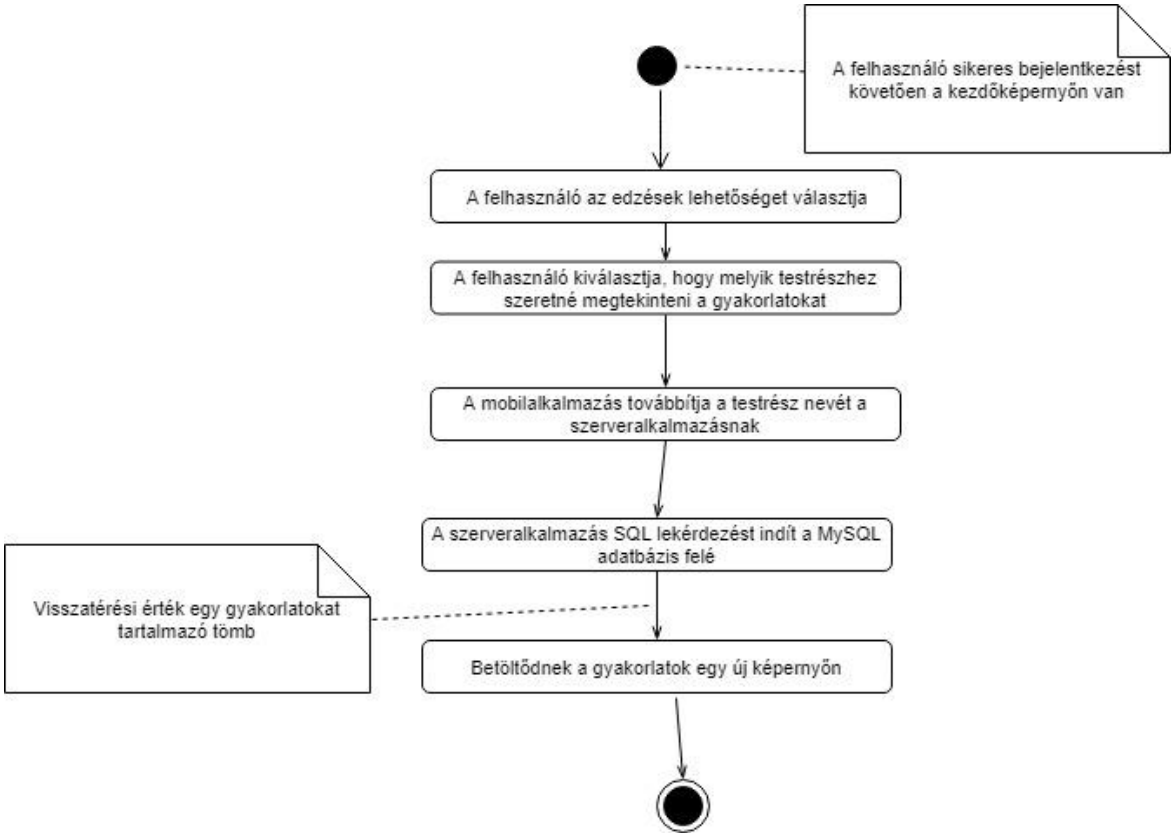
2.3. ábra: A bejelentkezés folyamat Activity diagramja

2.3.3 Gyakorlatok megjelenítése

A 2.5-ös ábrán a gyakorlatok megjelenítését tekinthetjük meg. A felhasználót sikeres bejelentkezést követően a kezdőképernyő fogadja. Itt két lehetőség közül választhat, de jelen részben csak az edzések lehetőséget fejtem ki. Miután kiválasztásra került az edzések opció láthatóak, hogy milyen testrészekhez vannak gyakorlatok az alkalmazásban. Ezek a testrészek a következők:

- mell,
- hát,
- comb,
- vádli,
- bicepsz,
- tricepsz,
- váll,
- has.

A felhasználó választása után, a testrész nevét küldi el a mobilalkalmazás a Szerveralkalmazás felé. A Szerveralkalmazás egy SQL lekérdezést indít a MySQL adatbázis felé, ahol feltételnek a testrész nevét adja meg. Visszatérési érték a Szerveralkalmazástól egy gyakorlatokat tartalmazó tömb lesz. Ezeket a gyakorlatokat fogja megjeleníteni egy új képernyőn a mobilalkalmazás.



2.4. ábra: A gyakorlatok megjelenítésének Activity diagramja

2.4 Kalóriaigény számítás

A napi szükséges kalóriabevitel kiszámításához a *Mifflin-ST. Jeor* képletet [4] használtam fel. A kalóriaszükségletnek a mértékegysége a *KJ (kilójoule)* pont, mint az energiának. Az emberi szervezet az energiát a bevitt tápanyagokból állítja elő. A kalóriaszámítás nemtől függően a következőképpen alakulhat:

- Férfiak esetén: (10 * súly) + (6.25 * magasság) – (5 * életkor) + 5
- Nők esetén: (10 * súly) + (6.25 * magasság) – (5 * életkor) – 161

Aktivitási szinttől függően a fent kapott értéket meg kell szorozni még a következőképpen:

Aktivitási szint	**Szorzó**
Könnyű	1,4
Közepes	1,7
Nehéz	1,9

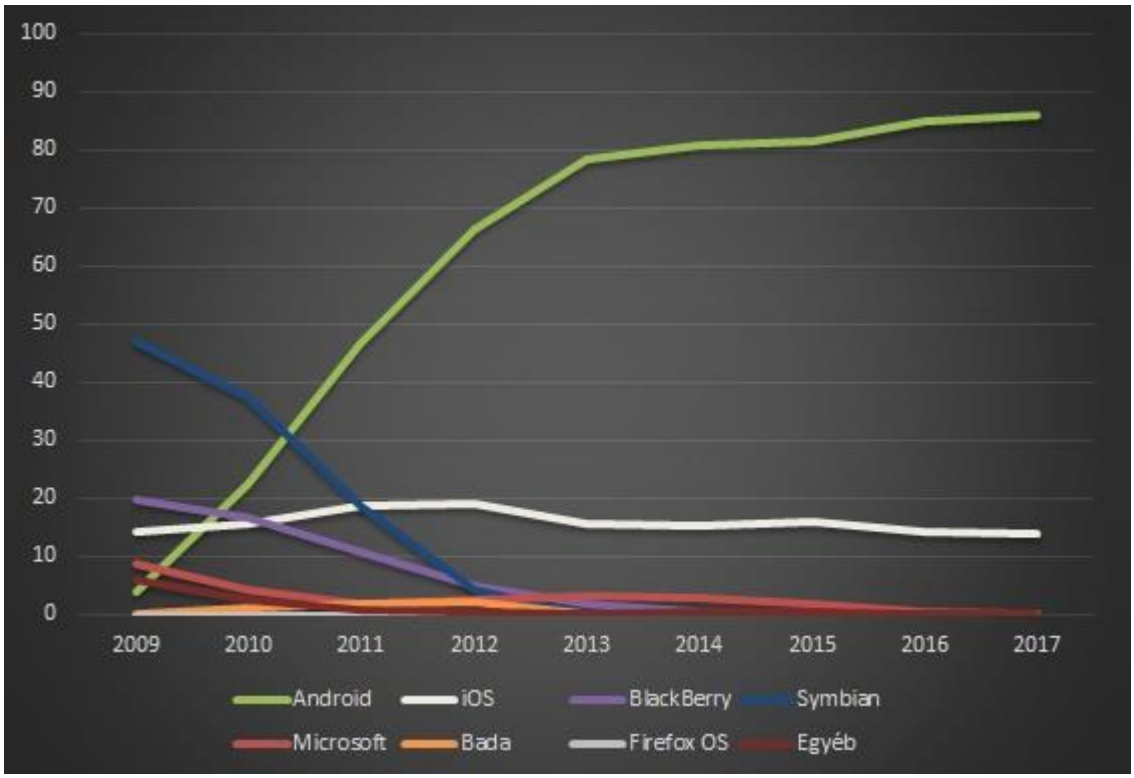
1. táblázat: Aktivitási szint szorzói

3 Felhasznált technológiák

A következőben azokat a technológiákat mutatom be, amelyeket felhasználtam az alkalmazás elkészítése során.

3.1 Android platform

Az Android alapja egy Linux operációs rendszer, amelyet átalakítottak úgy, hogy az eszközök integrált hardvereit megfelelően tudja kezelni. Az Android platform megalkotásának célja az volt, hogy egy nyílt forráskódú, rugalmas, könnyen alakítható rendszer legyen, amelyre könnyű külső alkalmazásokat fejleszteni. 2005-ben felvásárlásra került az Android Incorporated nevű kaliforniai cég, az „*IT óriás*” a Google által. 2007 elején kezdtek kiszivárogni olyan hírek, hogy a Google belép a mobil piacra és 2007. november 5-én az Open Handset Alliance bejelentette az Android platformot. Az első készülék a T-Mobile által forgalmazott, HTC G1-es készülék volt. Szakdolgozatom elkészítésekor az is az Android platform mellett szólt, hogy továbbra is a piaci részesedést hatalmas mértékben vezeti.



3.1. ábra: Mobilkészülékek piaci részesedésének az eloszlása [5]

3.2 Node.js

Skálázható internetes alkalmazások, méghozzá webszerverek készítésére hozták létre a Node.js [6] szoftverrendszert, amely lehetővé teszi JavaScript futtatását szerveroldalon is. A Node.js a Google-féle V8 JavaScript-motorból és számos beépített könyvtárból tevődik össze. Nem blokkoló I/O hívásoknak köszönhetően nagyon gyors, valamint sok hívás esetén is biztosítja, hogy az általunk használt szál ne legyen túlterhelt. Számomra ez megfelelő funkciókat látott el, ugyanis a nem blokkoló műveletek közé tartoznak olyanok, mint például:

- fájlműveletek,
- adatbázis műveletek.

Ezekre szükségem volt a szakdolgozatom elkészítése során, ugyanis sok adatott kellett kiszolgálnom.

3.3 Express

Az Express [7] egy nyílt forráskódú webalkalmazás keretrendszer a Node.js-hez. Felhasználásával sokkal könnyebben lehet a webalkalmazásunk kódját karbantartani és strukturálni.

3.4 MySQL

A MySQL egy relációs adatbáziskezelő rendszer, ahol az adatokat az adatbázis tábláiban tároljuk és ezek az eltárolt adatok innentől kezdve, az adatbázisnak egy-egy rekordjának felelnek meg. SQL utasítások segítségével tudjuk az adatokat kezelni. Ilyen utasítások lehetnek például:

- lekérdezések,
- adatok bevitele,
- adatok módosítása,
- adatok törlése.

3.4.1 Lekérdezések

```
SELECT (oszlopok)
FROM (táblák)
[WHERE (feltételek)]
[(csoportosítás)]
[(rendezés)];
```

A lekérdezés eredménye szintén egy táblát állít elő. Az *(oszlopok)* határozzák meg az eredménytábla oszlopait. A *(táblák)* adják a lekérdezésben résztvevő táblákat. A *(feltételek)* segítségével választhatjuk ki a számunkra szükséges rekordokat. A *(csoportosítás)* az eredménytábla sorait rendezi. A *(rendezés)* a megjelenítés sorrendjét befolyásolja.

3.4.2 Adatok bevitele

```
INSERT INTO (táblanév) [((oszlopnév) [, (oszlopnév), …])]
VALUES ((kifejezés1) [, (kifejezés2), …]);
```

Adatok bevitelekor, ha megadunk oszlopneveket, akkor minden megadott mezőnek értéket kell adni, a többi mező *null* értéket fog felvenni. Ha nem adunk meg oszlopneveket, akkor minden mezőnek értéket kell adni, különben parancsvégrehajtás után hiba üzenetet fogunk kapni.

3.4.3 Adatok módosítása

```
UPDATE (táblanév)
SET (oszlopnév) = (kifejezés) [, (oszlopnév) = (kifejezés2) , …]
[WHERE (feltételek)];
```

Amennyiben a *WHERE* feltételünk hiányozna, úgy az összes rekordot, ellenkező esetben csak a feltételeket teljesítő rekordok fognak módosulni az adatbázisunkban.

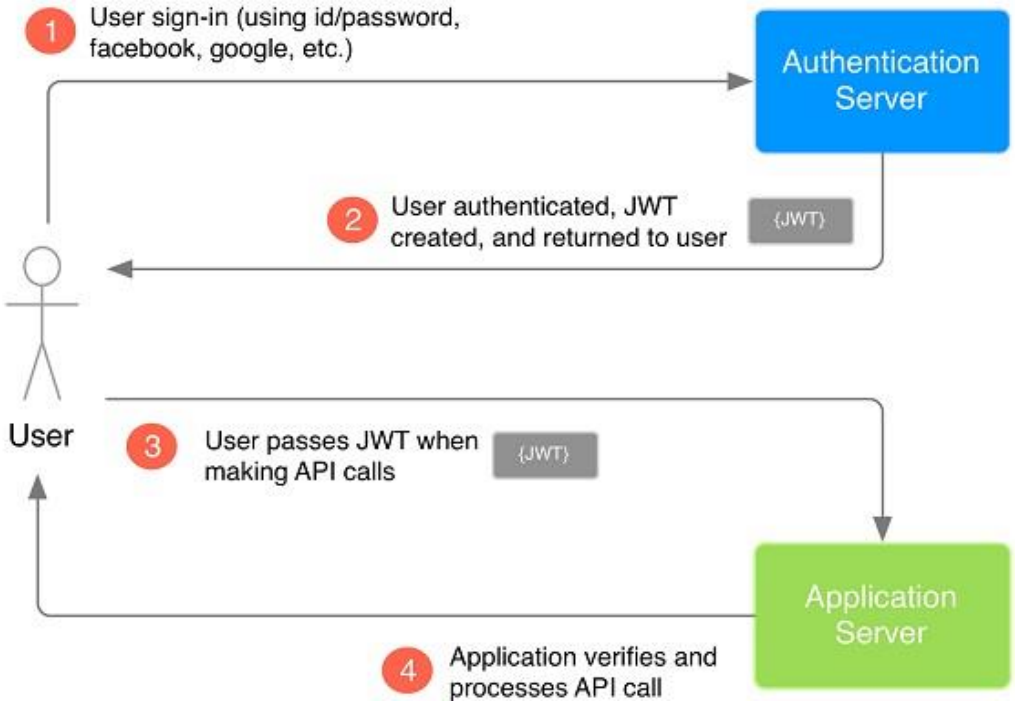
3.4.4 Adatok törlése

```
DELETE FROM (táblanév)
[WHERE (feltételek)];
```

Amennyiben a *WHERE* feltételünk hiányozna, úgy az összes rekordot, ellenkező esetben csak a feltételeket teljesítő rekordok fognak törlődni az adatbázisunkból.

3.5 JSON Web Token

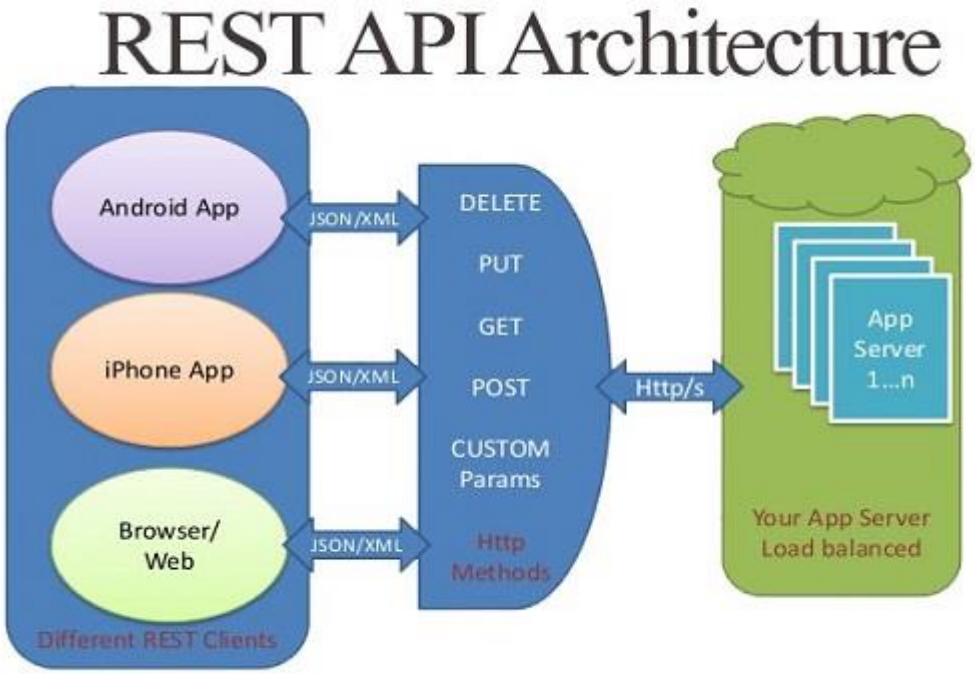
A JSON Web Token (JWT) [8] egy JSON objektum, amelyet az RFC 7519 szabvány definiál. A Token 3 fő részből tevődik össze, mégpedig fejléc, törzs és aláírás. Egy ilyen Token segítségével biztonságosan megvalósítható az információcsere két fél között, valamint bizonyos funkciók engedélyezése. Engedélyezésre egy kiváló példa a felhasználói bejelentkezés. Miután a felhasználó bejelentkezett, onnantól kezdve nem szükséges a számára, hogy *Id* és *Jelszó* alapján azonosítsa magát a szerver felé. Ha valami kérést indítana elegendő, ha a Token-t tartalmazza a kérés, ez alapján egyértelműen azonosítani lehet a felhasználókat. A Token rendelkezik még egy lejárati idővel, ami a Token érvényességének a dátumát fejezi ki.



3.2. ábra: JSON Web Token működése

3.6 REST

A REST (Representational State Transfer) [9] egy szoftverarchitektúra típus nagyobb internet alapú rendszerek számára. Az ilyen típusú architektúra szerverekből, valamint kliensekből épül fel. A kliensek különböző kéréseket tudnak küldeni a szerverek felé, amik a kéréseket feldolgozzák, és a választ továbbítják annak a kliensnek, aki a kérést indította.



3.3. ábra: REST API Architektúrája [10]

RESTful rendszernek nevezzük azokat a rendszereket, amelyek eleget tesznek a REST szabályainak. A szabályok a következők:

- állapotmentesség,
- kliens – szerver architektúra,
- réteges felépítés,
- gyorsítótárazhatóság,
- egységes interfész,
- igényelt kód (opcionális).

4 Architektúra bemutatása

Ebben a fejezetben az elkészült alkalmazás architektúrájának a felépítése kerül bemutatásra.

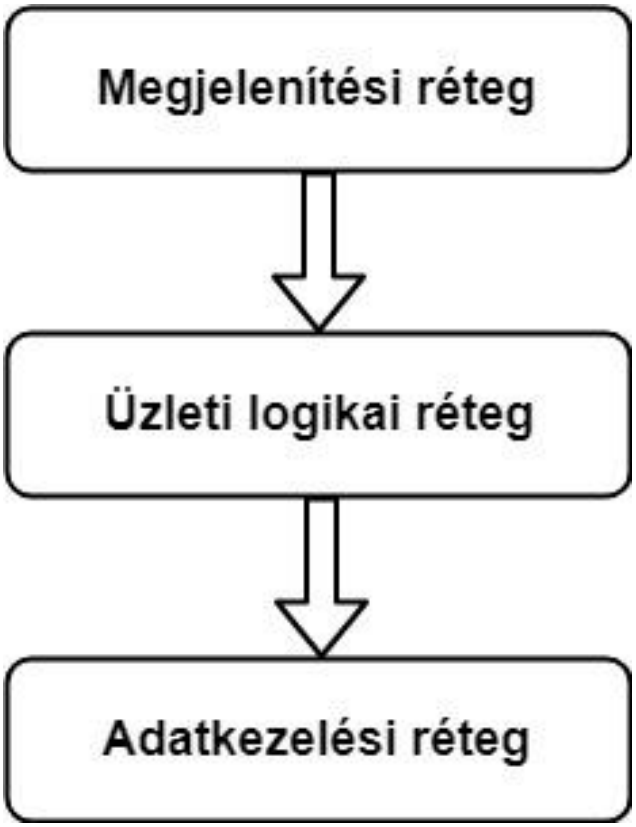
4.1 Háromrétegű architektúra

A többrétegű architektúrák a szoftverfejlesztésben előszeretettel alkalmazottak, ebben az esetben a teljes rendszerünk több különálló részből tevődik össze. A különálló rétegeknek köszönhetően számos előnyt biztosít számunkra más tervezési módszerekkel szemben, ilyenek például:

- Egyszerűbb tovább fejlesztési lehetőség biztosítása.
- Rétegek fejlesztése egymástól függetlenül is megvalósítható.
- Egy adott réteg akár teljes egészében kicserélhetővé válik.
- Lehetővé teszi a kód könnyebb karbantarthatóságát.

Ezen előnyöket figyelembe véve döntöttem úgy, hogy többrétegű architektúrát fogok használni a fejlesztésem során. Egyik legnépszerűbb a háromrétegű architektúra, amire pont illik az elkészített rendszerem. A háromrétegű architektúra rétegei:

- megjelenítési réteg,
- üzleti logikai réteg,
- adatkezelési réteg.



4.1. ábra: Háromrétegű architektúra felépítése

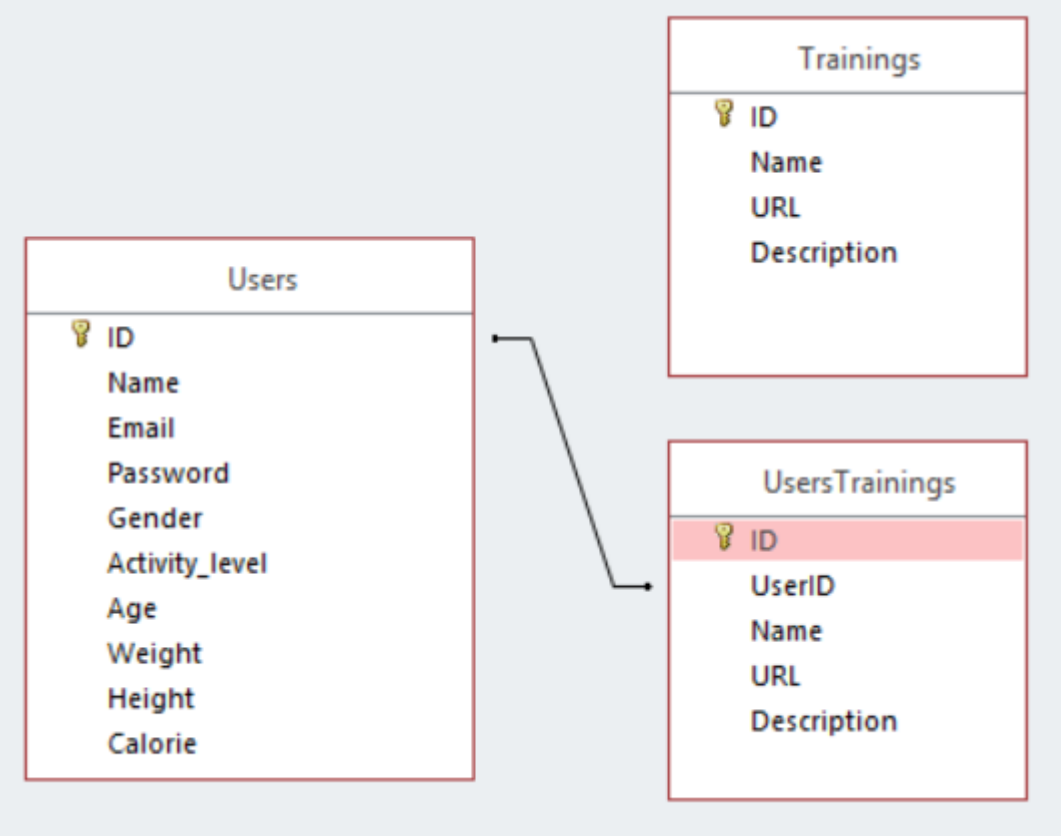
Amint a képen is látható nincs minden réteg összekötve a másikkal. Egy ilyen architektúrával készített rendszerben, a rétegek kizárólag a szomszédos rétegekkel képesek kommunikálni. Megjelenítési rétegnek az elkészült rendszeremben a mobilalkalmazás, Üzleti logikai rétegnek a szerveralkalmazás, Adatkezelési rétegnek pedig, a MySQL adatbázis felel meg.

5 Adatbázis

Ebben a fejezetben a MySQL adatbázisban elkészített táblákat fogom bemutatni, valamint a táblák között lévő kapcsolatokat.

5.1 Kapcsolatok a táblák között

Amint az 5.1-es ábrán látható, a *Users* tábla és a *Trainings* tábla között nincs semmilyen kapcsolat, azonban a *UsersTrainings* tábla, ennek a két táblának bizonyos adatainak az együtteséből tevődik össze.



5.1. ábra: Táblák közötti kapcsolat megjelenítése

5.2 Users tábla

Ebben a táblában a felhasználók adatait tárolom el, amiket a mobilalkalmazásban való regisztrációkor adnak meg. Az adatok típusáról, valamint további fontos tulajdonságaikról, a lentebb található táblázat nyújt segítséget.

Oszlop név	Típus	Kulcs	Üresen hagyható	Automatikus kitöltés
ID	INT(11)	Igen	Nem	Igen
Név	VARCHAR(45)	Nem	Nem	Nem
Email	VARCHAR(45)	Nem	Nem	Nem
Jelszó	VARCHAR(125)	Nem	Nem	Nem
Nem	VARCHAR(45)	Nem	Nem	Nem
Aktivitási szint	VARCHAR(45)	Nem	Nem	Nem
Életkor	INT(11)	Nem	Nem	Nem
Súly	INT(11)	Nem	Nem	Nem
Magasság	INT(11)	Nem	Nem	Nem
Kalória	DOUBLE	Nem	Igen	Nem

2. táblázat: Users tábla

5.3 Trainings tábla

Ebben a táblában a felhasználók által megtekinthető gyakorlatokat tárolom el. A gyakorlatok GIF-ek formájában tekinthetőek meg a mobilalkalmazásban. Ezeken a GIF-ken én szerepelek, én mutatom be a gyakorlatok szabályos kivitelezését.

Oszlop név	Típus	Kulcs	Üresen hagyható	Automatikus kitöltés
ID	INT(11)	Igen	Nem	Igen
Név	VARCHAR(45)	Nem	Nem	Nem
URL	VARCHAR(250)	Nem	Nem	Nem
Leírás	VARCHAR(500)	Nem	Nem	Nem

3. táblázat: Trainings tábla

5.4 UsersTrainings tábla

Ebben a táblában a felhasználók által kiválasztott gyakorlatokat tárolom el. Ezeket a gyakorlatokat a felhasználó a mobilalkalmazásban, saját edzésterveinél tudja megtekinteni.

Oszlop név	Típus	Kulcs	Üresen hagyható	Automatikus kitöltés
ID	INT(11)	Igen	Nem	Igen
UserID	INT(11)	Nem	Nem	Nem
Név	VARCHAR(45)	Nem	Nem	Nem
URL	VARCHAR(250)	Nem	Nem	Nem
Leírás	VARCHAR(500)	Nem	Nem	Nem

4. táblázat: UsersTrainings tábla

A *UsersID* attribútum egy külső kulcsnak felel meg, ami a felhasználót azonosítja. Ennek az attribútumnak a segítségével tudhatjuk, hogy adott felhasználó, milyen gyakorlatokat választott ki, amiket szeretne a saját edzéstervéhez adni.

6 Szerveralkalmazás

Ebben a fejezetben a Node.js-ben elkészített szerveralkalmazás felépítését, valamint fontosabb részeinek a működését fogom bemutatni.

6.1 Felépítés

A Szerveralkalmazás az alábbi komponensekből tevődik össze:

- controllers,
- counters,
- database,
- gifs,
- middleware:
    - validators.

A *LoginMiddleware* kezeli a felhasználók tokenjeit. Ezzel a tokennel tudják a felhasználók azonosítani magukat a szerver felé, vagyis ennek a tokennek minden olyan http kérés fejlécében szerepelnie kell, ami a felhasználóhoz kötött.

6.2 Controllers

A MySQL adatbázisban szereplő adatok lekérdezését, módosítását, valamint új adat felvételét teszik lehetővé. A szerveralkalmazás 3 ilyen kontrollerrel rendelkezik:

- userController,
- trainingController,
- usersTrainingsController.

6.2.1 UserController

A *UserController* osztálynak a feladata a felhasználók kezelése, ezért a *Users* táblához fér hozzá. Két fontos funkciót szeretnék itt megemlíteni, mégpedig a regisztrációt, és a bejelentkezést.

Regisztráció: Ez egy új adat felvételét jelenti a *Users* táblában. Adatok felvételének a folyamatát SQL nyelven már a 3.4.2-es részben részleteztem.

```
router.post('/', validator, calorie, (req,res) => {
  req.body.password = cryptoJS.SHA256(req.body.password);
  mysqlConnection.qurey('INSERT INTO users SET ?',
  req.body,(err, rows, fields) => {
        if(!err)
          res.json(rows)
        else
          res.json(err)
  });
});
```

A felhasználóknak nem a tényleges jelszava kerül eltárolásra, hanem annak egy hash-el előállított változata. Ez biztonság szempontjából fontos, mert ha jogosulatlan hozzáférés történne az adatbázishoz, akkor nem tudják kinyerni a támadók a felhasználók jelszavait.

Bejelentkezés: Ez egy lekérdezés valójában a *Users* táblában, ahol a megadott feltételt pontosan egy adat elégíti ki, vagy tévesen megadott adatok esetén egy sem. Adatok lekérdezésének a folyamatát SQL nyelven már a 3.4.1-es részben részleteztem.

```
router.post('/login', (req, res) => {
  var password = cryptoJS.SHA256(req.body.password).toString();
  mysqlConnection.query('SELECT * FROM users WHERE email = ?',
  req.body.email, function (err, rows, field) {
        if(rows.length > 0) {
          if(rows[0].Password == password) {
             var token = jwt.sign({
               exp: Math.floor(Date.now()) + (60 * 60 * 24 * 7 * 1000),
               data: row[0].ID
             }, secret);
             res.json({token: token, calorie: rows[0].Calorie});
          }
          else
             res.status(401).
             json({error: "Email jelszó páros nem jó!",
             attributeName: "emailPassword"});
        }
        else
          res.status(401).
          json({error: "Nincs ilyen email!",
          attributeName: "emailWrong"});
  });
});
```

6.2.2 TrainingController

A *TrainingController* osztálynak a feladata a gyakorlatok kezelése, ezért a *Trainings* táblához fér hozzá. Legfontosabb funkciója egy lekérdezés, ami egy gyakorlatokat tartalmazó tömböt ad vissza. Ennek alapján a mobilalkalmazásban betöltődnek az adott testrészhez megfelelő gyakorlatok.

Gyakorlatok lekérdezése: Ez több adat lekérdezése a *Trainings* táblából, ahol az adatok egy megadott feltételnek eleget tesznek.

```
router.get('/bodypart', (req, res) => {
  mysqlConnection.query('SELECT * FROM trainings WHERE name = ?',
  req.query.name, function (err, rows, fields) {
        if(rows.length > 0)
          res.json(rows);
        else
          res.status(401).
          json({error: „Nincs ilyen testrész!",
          attributeName: „nameEmpty"});
  });
});
```

6.2.3 UsersTrainingsController

A *UsersTrainingsController* osztálynak a feladata a felhasználó által kiválasztott gyakorlatok kezelése, ezért a *UsersTrainings* táblához fér hozzá. Két fontos funkciót szeretnék itt megemlíteni, mégpedig a gyakorlat felvételét, és a gyakorlat eltávolítását.

Gyakorlat felvétele: Ez egy új adat felvételét jelenti a *UsersTrainings* táblában.

```
router.post('/', loginMiddleware, validator, (req,res) => {
  mysqlConnection.qurey('INSERT INTO userstrainings SET ?',
  {UserID: req.userID, …req.body},(err, rows, fields) => {
        if(!err)
          res.json(rows)
        else
          res.json(err)
  })
});
```

Gyakorlat eltávolítása: Ez egy adat törlését jelenti a *UsersTrainings* táblában. Adatok törlésének a folyamatát SQL nyelven már a 3.4.4-es részben részleteztem.

```
router.post('/', loginMiddleware, (req,res) => {
  mysqlConnection.qurey('DELETE FROM userstrainings WHERE URL = ?',
  req.body.URL, function (err, rows, fields) {
        if(!err)
          res.json(rows)
        else
          res.json(err)
  });
});
```

6.3 Counters

Jelenleg az alkalmazás kizárólag a kalóriaszámításhoz tartalmaz egy számlálót. A 2.4-es résznél már ismertettem ennek a folyamatnak a számítási menetét, itt csak az implementációja valósult meg a képleteknek.

6.4 Database

```
const mysqlConnection = mysql.createConnection({
   host: 'localhost',
   user: 'root',
   password: 'Administrator'
   database: 'BuildBody'
   multipleStatements: true
});
```

A MySQL Adatbázishoz való csatlakozást biztosítja, méghozzá adminisztrátori jogosultságokkal.

6.5 Gifs

Ez a mappa tartalmazza a gyakorlatokat GIF fájlok formájában. Ennek a mappának az elérési útvonala, és adott gyakorlathoz tartozó GIF neve tárolódik el a *Trainings*, és a *UsersTrainings* táblák *URL* attribútumában.

6.6 Middleware

Jelenleg a Szerveralkalmazás csak a *LoginMiddleware*-rel rendelkezik. Ennek az osztálynak a feladata az alkalmazás indulásakor a token validációja. A validáció három fő részre osztható:

- Rendelkezésre áll a token.
- Lejárt a token.
- Módosult a token.

Amennyiben ezek a feltételek teljesülnek, a bejelentkezés sikeresen megtörténik.

6.7 Validators

A validátorok feladata az adatbázisba történő adatok felvételekor ellenőrizni, hogy a következő megkötések teljesülnek-e:

- Ki van töltve minden szükséges mező.
- Attribútum által elvárt típusban lettek kitöltve.
- Nem szerepel még az adatbázisban.

Amennyiben ezek a feltételek teljesülnek, az új adat sikeresen bekerül a MySQL adatbázisba. Jelenleg a Szerveralkalmazás három ilyen validátorral rendelkezik, amelyek a következők:

- userRegisterValidator,
- trainingRegisterValidator,
- usersTrainingsRegisterValidator.

7 Mobilalkalmazás

Ebben a fejezetben az Android alkalmazás felépítését, valamint fontosabb részeinek a működését fogom bemutatni.

7.1 Felépítés

Az Android alkalmazás az alábbi komponensekből tevődik össze:

- activitys,
- adapters,
- application,
- fragments,
- userInformation,
- recyclerViewElements.

7.2 Activitys

Feladatuk a Fragmentek eltárolása és egységes vezérlése. Implementálják a Fragmentek-ben elkészített interface-ket, amelyek Fragment váltáskor hívódnak meg. Van azonban egy Activity, aminek más feladata van, mégpedig a token validációja. Három Activity készült és hozzájuk az alábbi Fragmentek tartoznak:

- tokenValidationActivity,
- loginActivity:
    - loginFragment,
    - registrationFragment,
- mainActivity:
    - welcomeFragment,
    - bodyPartsFragment,
    - trainingsFragment,
    - myTrainingsFragment.

7.3 Adapters

RecyclerView elemeinek a tárolása valamint megjelenítése a feladata. Három ilyen adapter van:

- bodyPartsListAdapter,
- trainingsListAdapter,
- myTrainingsListAdapter.

7.4 Application

Applikációval kapcsolatos beállítások, hibakezelés, valamint Szerveralkalmazás felé indított kérések implementálását tartalmazza. Öt osztály tartozik ide:

- apiClient,
- apiInterface,
- errorUtil,
- prefConfig,
- validationError.

ApiCient: Localhost elérési címének és a retrofit2-nek a beállítása a feladata.

ApiInterface: Definiálja a http kéréseket a Szerveralkalmazás felé, hogy mit várnak válasznak és milyen értékeket várnak azok el.

ErrorUtil: *ValidationError* felhasználásával a hibakezelés megvalósításáért felelős.

Prefconfig: Kontextus, valamint a token beállítása a feladata. Elérhetővé teszi a token lekérdezését és írását.

ValidationError: A hibakezelésre lett létrehozva. Definiálja a hibák attribútumait.

7.5 Fragments

7.2-es részben már megemlítettem, hogy milyen Fragmentek-kel rendelkezik az alkalmazás. Ebben a részben az egyes Fragmentek feladatait részletezem.

LoginFragment: Bejelentkező képernyőért a felelős. Http kérést indít a Szerveralkalmazás felé bejelentkezés során, valamint validációt is végrehajt a megadott adatokon. Sikeres bejelentkezés esetén a *WelcomeFragment*-re, regisztráció esetén a *RegistrationFragment*-re továbbítja a felhasználót.

RegistrationFragment: Regisztrációs képernyőért felelős. Http kérést indít a Szerveralkalmazás felé regisztráció során, valamint validációt is végrehajt a megadott adatokon. Sikeres regisztráció esetén a *LoginFragment*-re továbbítja a felhasználót.

WelcomeFragment: Az alkalmazás kezdőképernyőjéért felelős. A felhasználó itt választhatja ki, hogy saját edzésterveit, vagy az edzéseket szeretné megtekinteni. Mindkét esetben a *BodyPartsFragment* fogadja a felhasználót, de annak tudatában, hogy ide honnan érkezett, már a *MyTrainingsFragment*, vagy a *TrainingsFragment* fogja fogadni.

BodyPartsFragment: A testrészeket lehet itt megtekinteni. Amennyiben kiválasztottuk a kívánt testrészt, a *WelcomeFragment*-nél említett módon, továbbít a következő Fragment-re.

TrainingsFragment: Adott testrészhez tekinthetőek meg itt a gyakorlatok, valamint hozzájuk tartozó leírás.

MyTrainingsFragment: Saját edzéstervhez felvett gyakorlatok tekinthetőek meg az adott testrésznél, valamint hozzájuk tartozó leírás.

7.6 UserInformation

Feladata a felhasználóval kapcsolatos információk kezelése, és tárolása. Három osztály tartozik ide:

- user,
- loginData,
- loginUserDto.

***User* feladata:** A felhasználó attribútumainak a definiálása. Továbbá lehetővé teszi, ezen attribútumok beállítását, és lekérdezését.

***LoginData* feladata:** Bejelentkezés esetén, a Szerveralkalmazástól érkező válasz feldolgozása, értékeinek az eltárolása.

***LoginUserDto* feladata:** Bejelentkezéskor megadott, email – jelszó páros eltárolása.

7.7 RecyclerViewElements

Feladata a RecyclerView elemeinek a definiálása, hogy azok milyen attribútumokkal rendelkeznek. Továbbá lehetővé teszi, ezen attribútumok beállítását, és lekérdezését. Két ilyen osztály van:

- bodyPart,
- training.

7.8 Kérés a Szerveralkalmazás felé

Ebben a részben a regisztrációt mutatom be, hogy hogyan is néz ki egy kérés indítása a mobilalkalmazásból, a Szerveralkalmazás felé.

***ApiInterface*-ben definiált kérés:**

```
@POST(”users”)
Call<User> performRegistration(@Body User user);
```

***RegistrationFragment*-ben megvalósítva:**

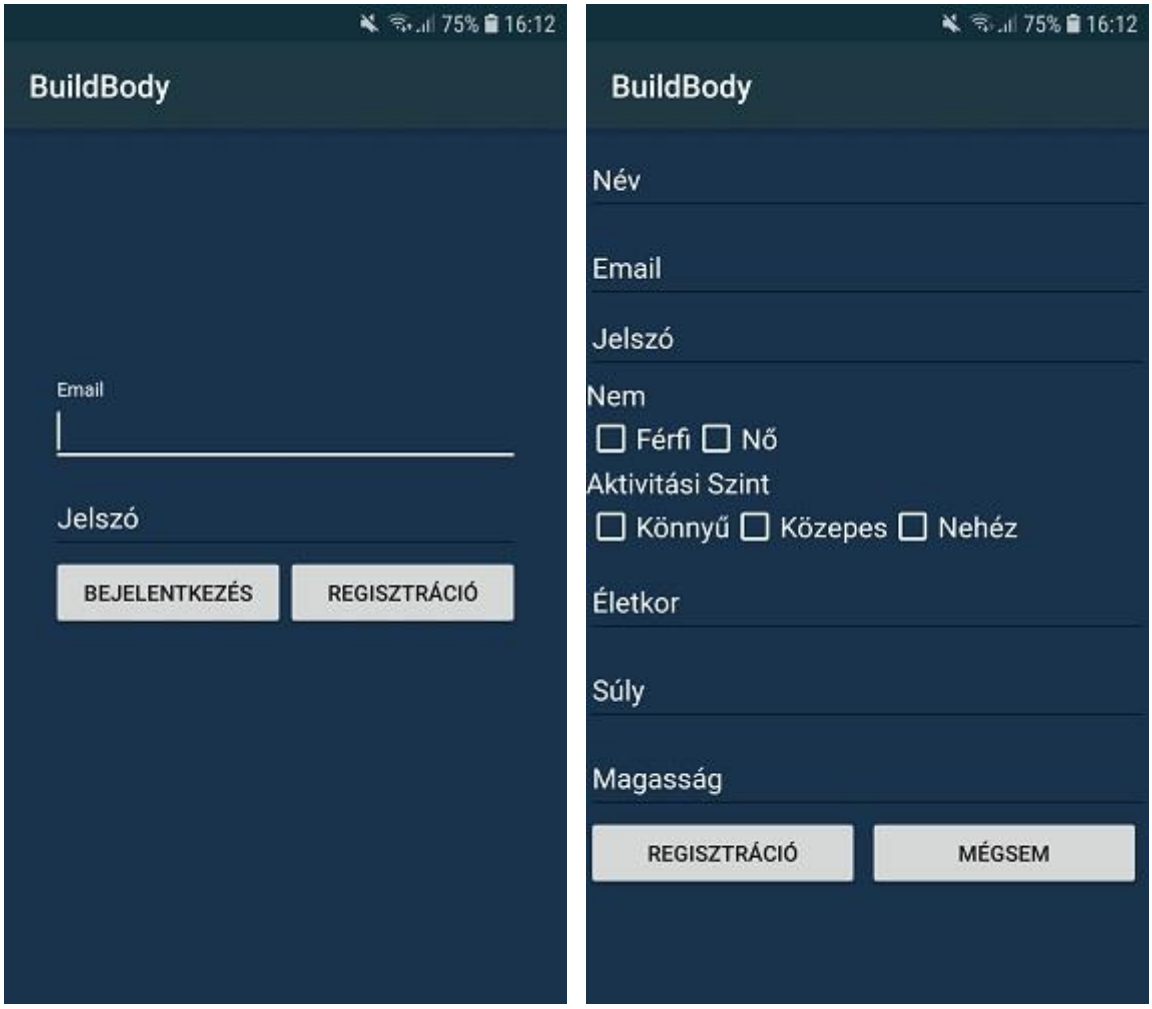
```
Call<User> call = LoginActivity.apiInterface.performRegistration(user);

call.enqueue(new Callback<User> (){
  @Override
  public void onResponse(Call<User> call, Response<User> response){
        if(response.isSuccesful()){
          registrationListener.applyPerformed();
        }
        else if(response.code()==403){
          ValidationError validationError = ErrorUtil.parseError(response);
          if(validationError.getAttributeName().equels(”email”)){
             emailTextInputLayout.setError(validationError.getError());
          }
        }
  }

  @Override
  public void onFailure(Call<User> call, Throwable t) {
        t.printStackTrace();
  }
});
```

8 Mobilalkalmazás használatának a bemutatása

Ahhoz, hogy használni tudjuk a mobilalkalmazást be kell jelentkezni. Amennyiben nem rendelkezünk még felhasználóval, abban az esetben ezt megelőzi egy regisztrációs folyamat. Regisztráció során minden adat kitöltése kötelező, más különben sikertelen a regisztráció.

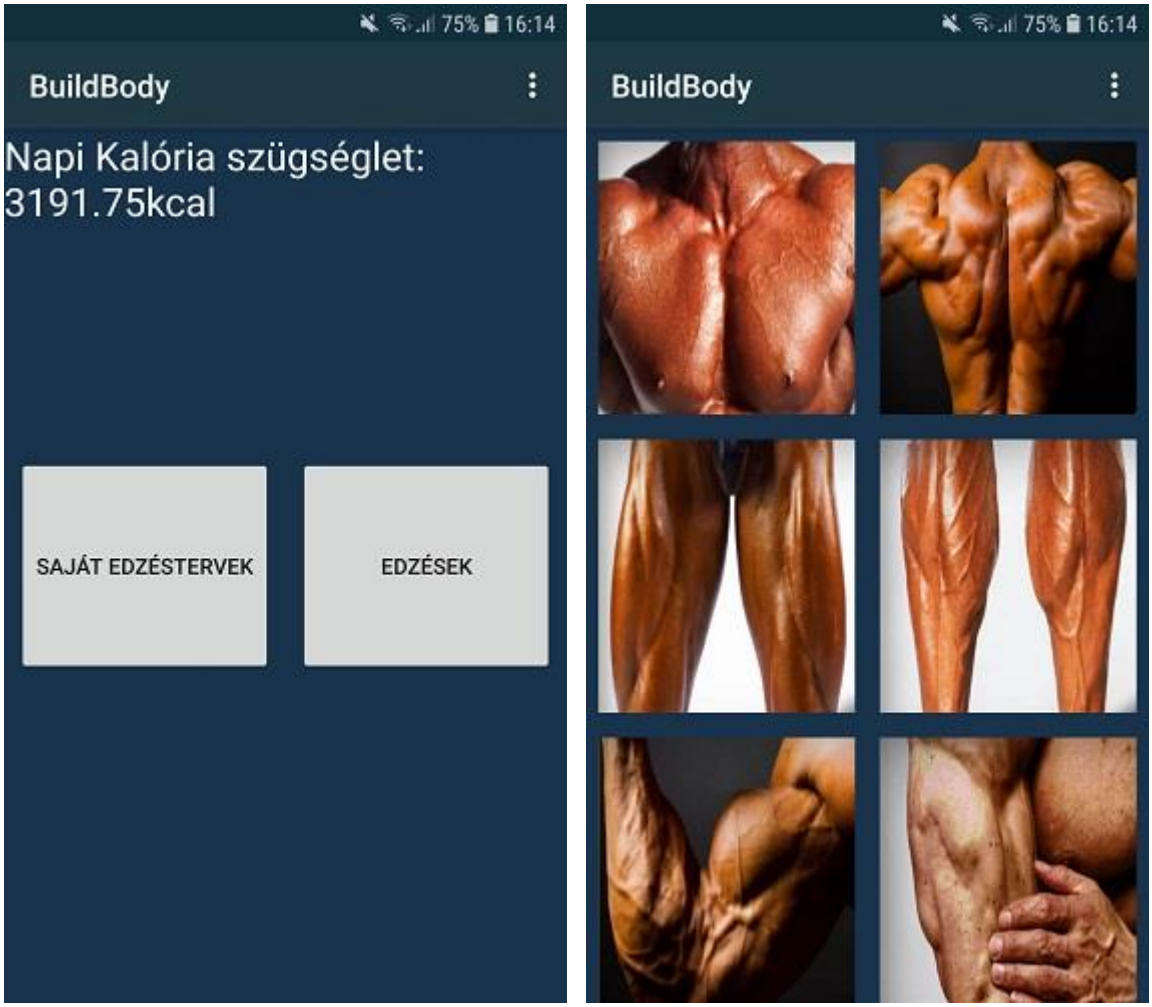


8.1. ábra: Bejelentkezés és regisztráció

Sikeres bejelentkezést követően a kezdőképernyő fogad minket. A képernyő tetején a napi ajánlott kalóriamennyiséget láthatjuk, továbbá két lehetőség közül választhatunk. Ez a két lehetőség a következő:

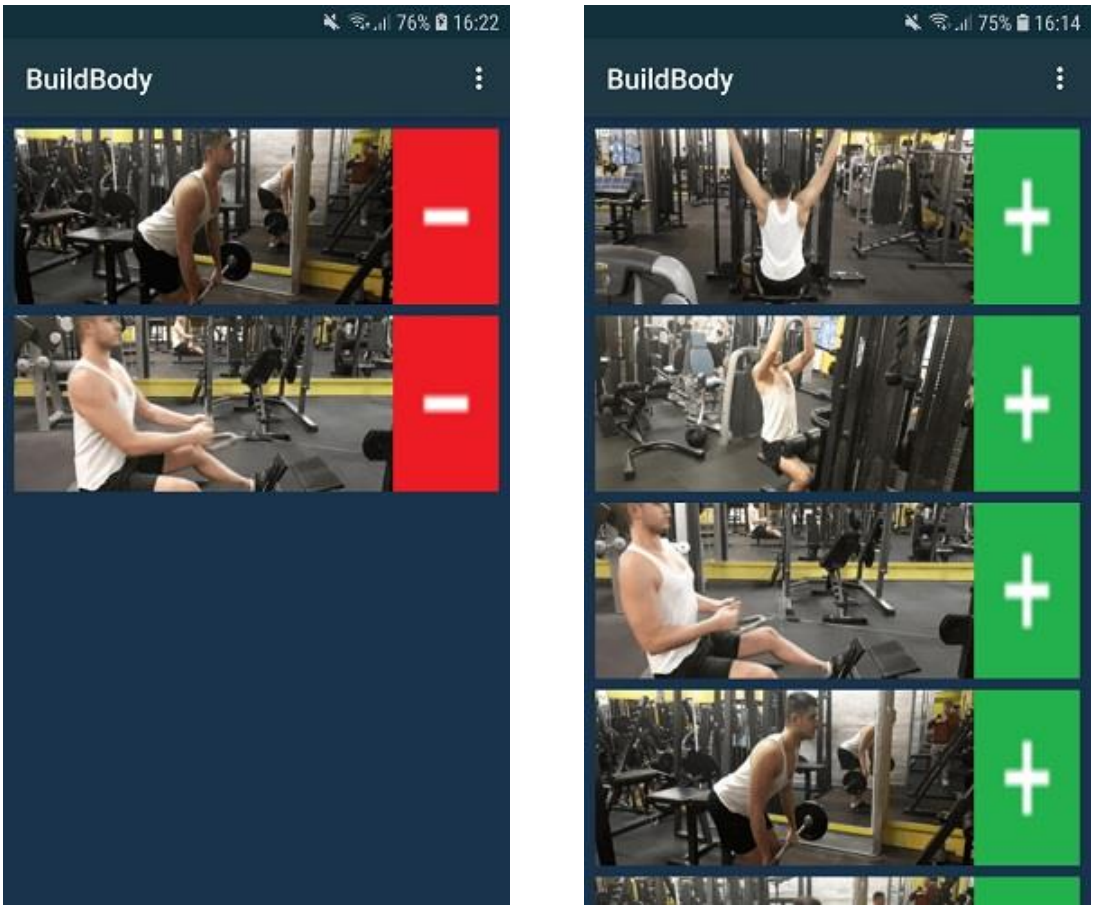
- saját edzéstervek,
- edzések.

Bármelyiket választjuk a két lehetőség közül, a következő képernyő mindkét esetben megegyezik. Ezen a képernyőn választhatjuk ki, hogy melyik testrészhez szeretnénk megtekinteni a gyakorlatokat.



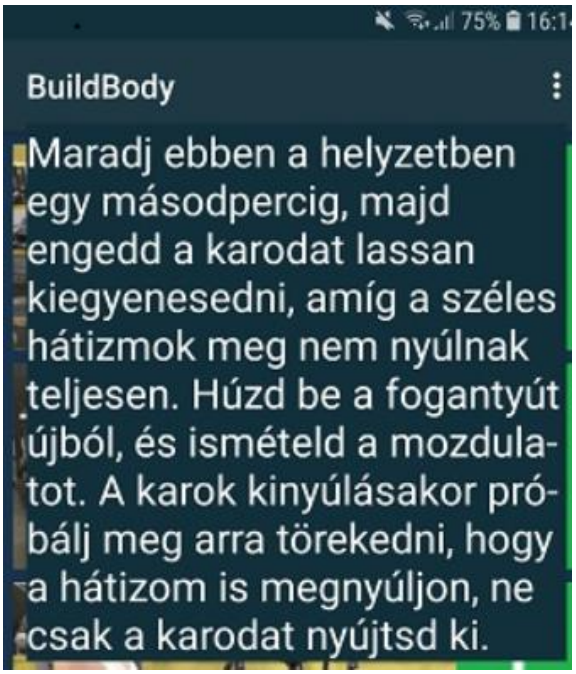
8.2. ábra: Kezdőképernyő

Attól függően, hogy a saját edzésterveket, vagy az edzéseket választottuk a kezdőképernyőn gyakorlatok megtekintésénél már eltérést találunk. A saját edzésterveknél a kiválasztott testrésznél, már csak azok a gyakorlatok lesznek, amiket mi hozzáadtunk. Megjelenítésben annyi különbséget tapasztalunk, hogy itt eltávolítás gomb van a gyakorlat mellett, nem pedig hozzáadás gomb.



8.3. ábra: Saját edzések és gyakorlatok

Ezen kívül, ha többet szeretnénk megtudni egy gyakorlatról, akkor a GIF-re kattintva Popup ablakban, rövid leírást olvashatunk a gyakorlat kivitelezéséről.



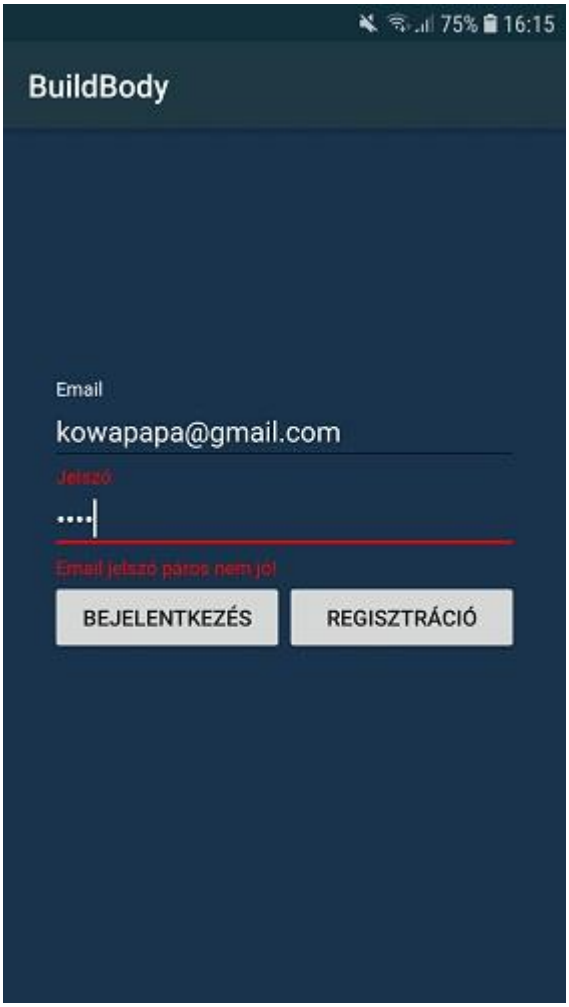
8.4. ábra: Gyakorlat leírásának a megjelenítése

9 Tesztelés

Ahhoz, hogy szakdolgozatom igazi mérnöki munka legyen, elengedhetetlen részét képezi, ennek az egész rendszernek a tesztelése.

9.1 Mobilalkalmazás tesztelése

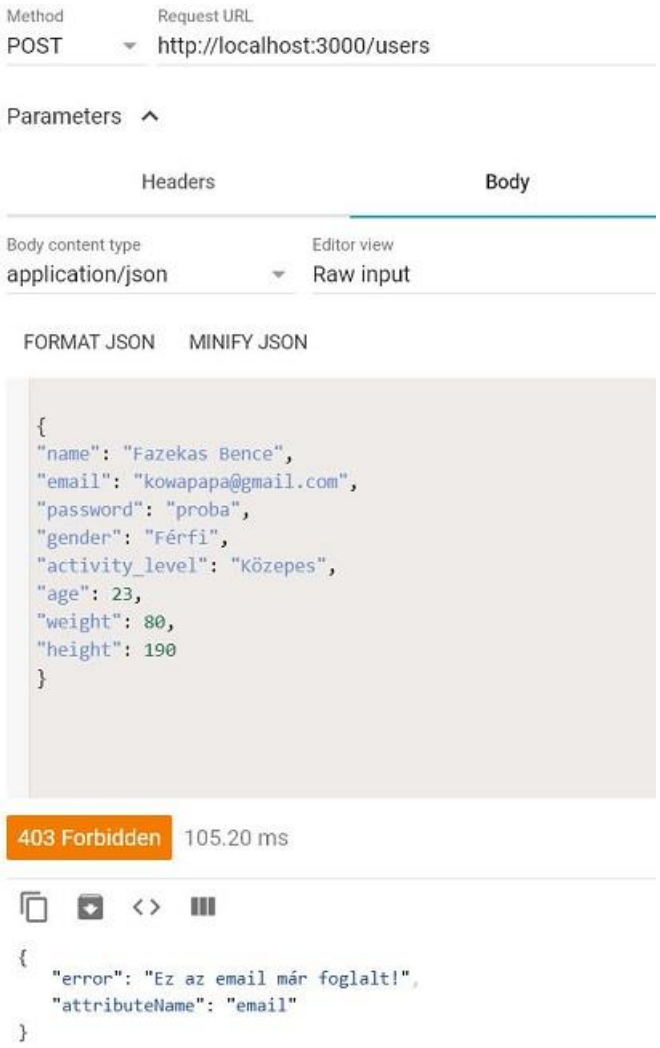
A mobilalkalmazás tesztelését az egész fejlesztés során szem előtt tartottam, hogy ne a végén derüljön ki, hogy bizonyos részek nem az elvárt működést produkálják. A Szerveralkalmazástól érkező hibák kezelésére külön osztályt hoztam létre, erről a 7.4-es résznél részletesebben is írtam. Különféle státuszkódú válaszok feldolgozását is figyelembe veszem, erre példát a 7.8-as résznél lehet megtekinteni, ahol a regisztráció menetét mutattam be.



9.1 ábra: Email jelszó páros hibás

9.2 Rest API tesztelése

Tesztelésére a regisztrációt mutatom be, annak is azt az esetét, amikor a megadott email címmel már létezik felhasználó az adatbázisban. A tesztelésre az *Advanced REST client* nevű alkalmazást használtam. A fejlesztés során a kéréseket többször is teszteltem, hogy biztos legyek abban, hogy az elvárt működés szerint zajlik a folyamat.



9.2 ábra: Az email cím már foglalt

10 Összegzés és továbbfejlesztési lehetőségek

Az utolsó fejezetben összefoglalom a fejlesztés folyamatát és kitérek a továbbfejlesztési lehetőségekre.

10.1 Fejlesztési folyamat összefoglalása

A szakdolgozatom célja egy olyan Android alkalmazás fejlesztése volt, amely segítségével a felhasználók meg tudják tekinteni különböző izomcsoportokhoz tartozó gyakorlatokat, hogyan is kell szabályosan elvégezni. Megadott adataik alapján tisztában vannak vele, hogy napi szinten, mennyi az ajánlott kalóriabevitel számukra, továbbá lehetőségük van saját edzéstervet létrehozni.

Ahhoz, hogy ez a rendszer elkészüljön és működőképes legyen, készítenem kellett egy MySQL adatbázist ahol az adatok tárolását valósítottam meg. Egy Szerveralkalmazásra is szükségem volt, amin keresztül hozzáférek az adatbázisomban tárolt adatokhoz, és kéréseket indíthatok felé. Végül pedig, az Android alkalmazás a felhasználók számára a fent említett funkciókkal.

Úgy érzem szakdolgozatom céljait sikeresen teljesítettem, és számos új ismeretet szereztem mind a webfejlesztés, mind az Android-os mobilkészülékekre való fejlesztésben.

10.2 Továbbfejlesztési lehetőségek

A fejlesztés során, amikor a gyakorlatokat készítettem el, felmerült bennem az ötlet, hogy esetleg a felhasználóknak is kéne egy olyan funkciót biztosítani, amivel ők is tudnak gyakorlatokat felvenni az adatbázisba. Ezeket a gyakorlatokat más felhasználók is meg tudnák tekinteni, és amennyiben elnyeri tetszésüket a saját edzéstervükhöz fel tudnák venni.

Egy másik fejlesztési lehetőség pedig, hogy az alkalmazás támogassa a Google, valamint a Facebook bejelentkezést is. Ilyen bejelentkezés esetén azonban, első bejelentkezéskor meg kell adnunk a napi kalóriamennyiség számításához szükséges adatokat.

11 Irodalomjegyzék

[1] [Online]. Available: https://play.google.com/store/apps/details?id=sixpack.sixpackabs.absworkout. [Hozzáférés dátuma: 2018.09.10.].

[2] [Online]. Available: https://play.google.com/store/apps/details?id=armworkout.armworkoutformen.armexercises. [Hozzáférés dátuma: 2018.09.10.].

[3] „Diagram készítés," [Online]. Available: https://www.draw.io/. [Hozzáférés dátuma: 2018.11.20.].

[4] „Kalória számítás," [Online]. Available: https://czinadora-hu.webnode.hu/products/alapanyagcsere-es-napi-energiaszukseglet-kiszamitasa/. [Hozzáférés dátuma: 2018.11.26.].

[5] H. Ferenc, „Mobilkészülékek piaci részesedésének az eloszlása," [Online]. Available: https://www.hwsw.hu/hirek/58482/android-ios-okostelefon-platform-gartner.html. [Hozzáférés dátuma: 2018.11.19.].

[6] „Node.js," [Online]. Available: https://nodejs.org/en/about/. [Hozzáférés dátuma: 2018.11.20.].

[7] „Express," [Online]. Available: https://expressjs.com/. [Hozzáférés dátuma: 2018.11.20.].

[8] „JSON Web Token működése," [Online]. Available: https://medium.com/vandium-software/5-easy-steps-to-understanding-json-web-tokens-jwt-1164c0adfcec. [Hozzáférés dátuma: 2018.11.20.].

[9] „REST," [Online]. Available: https://hu.wikipedia.org/wiki/REST. [Hozzáférés dátuma: 2018.11.20.].

[10] [Online]. Available: https://shareurcodes.com/photos//rest-api.jpg. [Hozzáférés dátuma: 2018.11.20.].