

**VÁCI SZAKKÉPZÉSI CENTRUM
BORONKAY GYÖRGY
MŰSZAKI TECHNIKUM ÉS GIMNÁZIUM**

SZAKDOLGOZAT

**Bakoss Borka
Dutka Lehel Ákos
Ondrik András**

2023.

VÁCI SZAKKÉPZÉSI CENTRUM
BORONKAY GYÖRGY
MŰSZAKI TECHNIKUM ÉS GIMNÁZIUM



SZAKDOLGOZAT

Fitclock

Konzulens: Kemenes Tamás
Wiezl Csaba

Készítette: Bakoss Borka
Dutka Lehel Ákos
Ondrik András

Hallgatói nyilatkozat

Alulírottak, ezúton kijelentjük, hogy a szakdolgozat saját, önálló munkánk, és korábban még sehol nem került publikálásra.

Bakoss Borka

Dutka Lehel Ákos

Ondrik András

Konzultációs lap

Vizsgázók neve: Bakoss Borka, Dutka Lehel Ákos, Ondrik András

Szakdolgozat címe: Fitclock

Program nyújtotta szolgáltatások:

- Edzéstervek készítése
- Egyéni edzési idők
- Egyéni fejlődés nyomon követése
- Kihívások küldése
- Visszajelzések kiértékelése
- Felhasználói statisztikák
- Adminisztrációs felügyelet

Sorszám	A konzultáció időpontja	A konzulens aláírása
1.	2022.10.10.	
2.	2022.11.14.	
3.	2022.12.12.	
4.	2023.01.16	
5.	2023.02.13.	
6.	2023.03.13	

A szakdolgozat beadható:

Vác, 2023.

A szakdolgozatot átvettem:

Vác, 2023.

Konzulens

A szakképzést folytató
intézmény felelőse

Tartalomjegyzék

Hallgatói nyilatkozat.....	2
Konzultációs lap.....	3
Témaválasztás.....	6
1 Fejlesztői dokumentáció.....	7
1.1 Technológiák.....	7
1.1.1 Windows Forms és a .Net Framework.....	7
1.1.2 ML.NET.....	7
1.1.3 MySQL és Apache.....	7
1.1.4 Composer.....	7
1.1.5 PHP.....	8
1.1.6 Javascript.....	8
1.1.7 Twig.....	8
1.1.8 Xamarin.....	8
1.2 Fejlesztői környezetek.....	9
1.2.1 Microsoft Visual Studio 2019.....	9
1.2.2 Visual Studio Code.....	9
1.3 Fejlesztés menete.....	9
1.4 Az adatbázis felépítése.....	10
1.4.1 Táblák kapcsolata.....	10
1.4.2 A táblák.....	11
1.5 Jogosultságok.....	20
1.5.1 Feladatkörök.....	20
1.5.2 Jogosultsági szintek.....	21
1.6 Az asztali alkalmazás felépítése.....	21
1.6.1 Model.....	21
1.6.2 ViewModel.....	21
1.6.3 View.....	21
1.6.4 A felületek.....	22
1.6.5 A tanítás menete.....	23
1.6.6 A kerekített ablakok.....	24
1.6.7 A regisztráció menete.....	24
1.7 A mobil alkalmazás felépítése.....	25
1.7.1 Images.....	25
1.7.2 MarkupExtensions.....	25
1.7.3 Model.....	25
1.7.4 Renderer.....	25
1.7.5 Services.....	25
1.7.6 Felületek.....	26
1.8 A weboldal felépítése.....	30
1.8.1 App.....	30
1.8.2 Vendor.....	31

1.8.3 index.php.....	31
1.8.4 newpic.jpg és newProfPic.jpg.....	31
1.8.5 A felületek.....	32
1.9 Tesztelés.....	39
1.9.1 Az asztali alkalmazás tesztjei.....	39
1.10 Továbbfejlesztési lehetőségek.....	42
1.10.1 Az asztali alkalmazás.....	42
1.10.2 A mobil alkalmazás.....	43
1.10.3 A weboldal.....	44
2 Felhasználói dokumentáció.....	45
2.1 Az asztali alkalmazás.....	45
2.1.1 Rendszerkövetelmények.....	45
2.1.2 Telepítési útmutató.....	45
2.1.3 Felületek.....	46
2.2 A mobil alkalmazás.....	49
2.2.1 Rendszerkövetelmények.....	49
2.2.2 Telepítési útmutató.....	49
2.2.3 Felületek.....	49
2.3 A weboldal.....	52
2.3.1 Rendszerkövetelmények.....	52
2.3.2 Telepítési útmutató.....	52
2.3.3 Felületek.....	53
3 Irodalomjegyzék.....	57
4 Mellékletek.....	59

Témaválasztás

A témánkat egy mobilos alkalmazás ötlete adta, mely András által észrevett problémát old meg.

Jelenleg a piacon nem található olyan időzítő, amely képes lenne körönként más idővel folyamatosan lefutni, ezért ezt akartuk elérni és e köré építettünk egy alkalmazás csoportot.

A mobilos alkalmazást András készítette el. Az alkalmazás elsődleges funkciója az időzítő, de ezen felül lehet saját edzésterveket készíteni, értékelni rendszeren belül. Az időzítőn belül kiválaszthatjuk a gyakorlatot, így pontos adatokat képes tárolni az edzésünkkel kapcsolatban. Az alkalmazásba való bejelentkezéssel további funkciókat lehet elérni például statisztikák megtekintése weboldalon keresztül.

A weboldalt Lehel készítette el. A weboldal használatához szükséges fiókkal rendelkezni. Itt tekinthetők meg az edzések statisztikái, javaslatokat kaphatunk gyakorlatokra, attól függően, hogy milyen izomcsoporton szeretnénk dolgozni. Csoportokat lehet létrehozni, amivel további funkciókat érhetünk el, például csoport tagjairól is nézhetünk statisztikai diagramokat. Az egyes csoportok adminjai még több funkcióhoz férhetnek hozzá.

Mindkét rész egyszerű és felhasználóbarát felülettel rendelkezik, emiatt minden korosztály számára ajánlott. Elsősorban a tanárok, edzők számára lehet segítség a csoportos funkció, mert így felügyelhetik, diákjaik, csoportjaik statisztikáit.

Az adatbázis felügyeletéhez tartozó asztali alkalmazást és az adatbázist Borka készítette el. Az alkalmazáshoz adminisztrátori jogosultság kell. Ahhoz, hogy valaki adminisztrátori fiókot készítsen, szükséges egy hitelesítési kulcsot megadni. a felületen láthatja a felhasználók értékeléseinek statisztikáit. Egy külön felületen láthatóak a szöveges értékelések automatikusan kategorizálva gépi tanulással. A szöveges értékelések alapján kiszámítja továbbá a felhasználók elégedettségét.

A könnyű használhatóság érdekében egyszerű, felhasználóbarát felülettel készült az alkalmazás, így még a technológiában nem jártas személyek is könnyen tudják kezelni az adminisztrációs felületet.

A dokumentáció közös részeit Borka írta meg, valamint mindenki a saját részéhez tartozó fejlesztői és felhasználói dokumentációt készítette el.

1 Fejlesztői dokumentáció

Készítette: Bakoss Borka, Dutka Lehel Ákos, Ondrik András

1.1 Technológiák

1.1.1 Windows Forms és a .Net Framework

A Windows Forms egy UI keretrendszer Windows asztali alkalmazások készítéséhez. A vizuális tervezője lehetővé teszi a produktív, gyors fejlesztést. Lehetővé teszi a grafikus elemekben gazdag alkalmazások fejlesztését. A Windows Forms alkalmazások hozzáférhetnek lokális fájlokhoz, vagy akár szerveren lévő fájlokhoz.

A .NET Framework tartalmazza a saját fejlesztő eszközeit és osztály könyvtárait. Többféle programozási nyelvvel használható és többféle platformra is készíthetünk vele alkalmazásokat, például asztali alkalmazást vagy mobilos alkalmazást.¹

1.1.2 ML.NET

Az ML.NET egy nyílt forráskódú machine learning keretrendszer a .NET fejlesztői platformhoz. Lehetővé teszi, hogy saját modellt építsünk fel és tanítsunk be. Olyan esetekben is használható, mint például vélemények analízise, ár megbecslése, termék ajánlása, kép besorolása, tárgy felismerése.²

1.1.3 MySQL és Apache

Az Apache HTTP szerver egy nyílt forráskódú HTTP server Windows és Unix alapú operációs rendszerek számára. Feldolgozza a kliens kéréseit és továbbítja a szerver válaszokat a HTTP-n keresztül.

A MySql egy relációs adatbázis kezelő rendszer, mely az adatokat táblákban tárolja el. Az adatok logikailag rendezve vannak, oszlopokba, sorokba, illetve egyes táblák között kapcsolatokat hozhatunk létre. A szabályokat a fejlesztő hozza létre, és elérheti, hogy az adatbázisban ne lehessenek duplikációk, nem megfelelő adatok, hiányzó adatok.³

1.1.4 Composer

A Composer egy alkalmazásszintű függőségkezelő a PHP programozási nyelvhez, amely szabványos formátumot biztosít a PHP szoftverek és a szükséges könyvtárak függőségének kezelésére.⁴

¹ Documentation Windows Forms: <https://learn.microsoft.com/en-us/dotnet/desktop/winforms/overview/?view=netdesktop-7.0>
Optimizely: What to know about .NET Framework: <https://www.optimizely.com/insights/blog/what-is-net-framework/>

² Microsoft What is ML.NET: <https://dotnet.microsoft.com/en-us/learn/ml-dotnet/what-is-ml-dotnet>

³ Apache documentation: https://httpd.apache.org/ABOUT_APACHE.html

Oracle What is MySQL <https://www.oracle.com/mysql/what-is-mysql/>

⁴ Composer Documentation: <https://getcomposer.org/doc/00-intro.md>

1.1.5 PHP

A PHP egy széleskörűen felhasznált, nyílt forráskódú, általános felhasználású szkriptnyelv dinamikus weboldalak fejlesztéséhez, mely beágyazható HTML kódba.

Elsősorban a szerveroldali szkriptnyelv, így képes adatot gyűjteni a kienstől, dinamikus elemeket készíteni, valamint sűtikek használni.

PHP támogatja az operációs rendszerek és webserverek többségét. Továbbá a PHP nyelv alkalmas az objektum orientált vagy a procedurális programozásra is.⁵

1.1.6 Javascript

A JavaScript egy objektumorientált, prototípus-alapú szkriptnyelv, amelyet weboldalakon elterjedten használnak. A Javascript által complex elemeket implementálhatunk a weboldalunkon. Lehetővé teszi az elemek dinamikus frissítését, multimédia vezérlését, képek animálását.

A kliens oldali Javascript a leggyakoribb, melyet vagy a HTML kódban helyeznek el, vagy onnan hivatkoznak rá.

Előnye, hogy kevesebb szerver interakciót igényel, azonnali visszajelzést küld, megnöveli az interakciók lehetőségeinek számát, és gazdagabb interfészeket tesz lehetővé.⁶

1.1.7 Twig

A Twig egy gyors, biztonságos és rugalmas PHP sablon feldolgozó szoftver. Könnyen kezelhető mind a fejlesztők, mind a tervezők számára azáltal, hogy ragaszkodik a PHP szabályaihoz, valamint a sablonkezelő környezeteknek funkcionalitást rendel hozzá.⁷

1.1.8 Xamarin

A Xamarin egy nyílt forráskódú platform modern alkalmazások építésére iOS, Android és Windows felületre .NET segítségével. Xamarin egy absztrakciós réteg, amely irányítja a megosztott kód kommunikációját az alap platform kódjával.

Xamarin lehetővé teszi a fejlesztők számára, hogy a teljes alkalmazás logikáját egyetlen nyelven, de natív működést, külsőt érjenek el különböző platformokon.⁸

⁵PHP Documentation: <https://www.php.net/manual/en/introduction.php>

⁶Tutorialspoint Javascript overview: https://www.tutorialspoint.com/javascript/javascript_overview.htm

⁷Twig Documentation: <https://twig.symfony.com/doc/3.x/intro.html>

⁸ Documentation Xamarin: https://learn.microsoft.com/hu-hu/xamarin/get-started/what-is-xamarin?WT.mc_id=dotnet-35129-website

1.2 Fejlesztői környezetek

1.2.1 Microsoft Visual Studio 2019

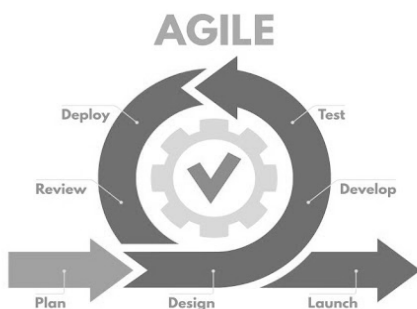
A Visual Studio IDE megfelelő környezet arra, hogy szerkeszd, debugold, illetve felépítsd a kódot, majd publikáld. Tartalmaz compilereket, kód kiegészítő eszközöket, grafikus tervezőket és még más eszközöket, hogy a szoftverfejlesztés menetét javítsa. Többféle programozási nyelvet támogat, mint például XML, C#, F#, C++ és Visual Basic.⁹

1.2.2 Visual Studio Code

Visual Studio Code egy ingyenes nyílt forráskódú kódszerkesztő. Alapértelmezetten támogatja a JavaScript, TypeScript és Node.js-t, de tartalmaz kiegészítőket, melyek letöltésével más nyelveket is képes feldolgozni, mint például a PHP, C# és Java.¹⁰

1.3 Fejlesztés menete

A fejlesztés során az agilis módszertannak megfelelően, ebből is az extreme programming (XP) módszertant követtük.



Az extreme programming gyors, hatékony fejlesztést tesz lehetővé rövid idő alatt, valamint alkalmas olyan esetekre mikor az igények dinamikusan változnak, vagy a rendszer működőképessége nem teljesen biztos.

Alapvetően 6 fázisa van:

- Tervezés (Planning)
- Elemzés (Analysis)
- Design
- Végrehajtás (Execution)
- Zárás (Wrapping)

Tervezésnél megbeszéltük a szoftverek szolgáltatásait, a biztonsági követelményeket.

Az elemzés szoros összefüggésben volt a tervezéssel. Megállapítottuk, hogy minek kell megfelelni a szoftvereknek, hogy a lehető legjobb felhasználói élményt nyújtsák, ezenkívül továbbá megállapítottuk, hogy milyen technológiák lennének alkalmasak a szoftverek megvalósítására.

A design során elosztottuk a különböző feladatokat, és mindenki megtervezte, hogy milyen tesztforgatókönyvek kellenek a saját feladataihoz.

Végrehajtás során pedig elkészítettük a szoftvereket folyamatos tesztelés mellett. Az egyes kódészeket egymásnak is teszteltük, ezáltal biztosítva, hogy megfelelőképp működjenek.

Zárásnál felállítottuk a rendszereket és elkészült a dokumentáció végleges verziója.¹¹

⁹ Microsoft Visual Studio: <https://visualstudio.microsoft.com/>

¹⁰ Visual Studio Code: <https://code.visualstudio.com/>

¹¹ Proman Consulting Mi az agilis módszertan: <https://promanconsulting.hu/mi-az-agilis-modszertan-legelterjedtebb-agilis-modszertanok/>

1.4 Az adatbázis felépítése

Adatbázis szerver típusa: MariaDB

Szerver verziója: 10.4.6

Webszerver típusa: Apache

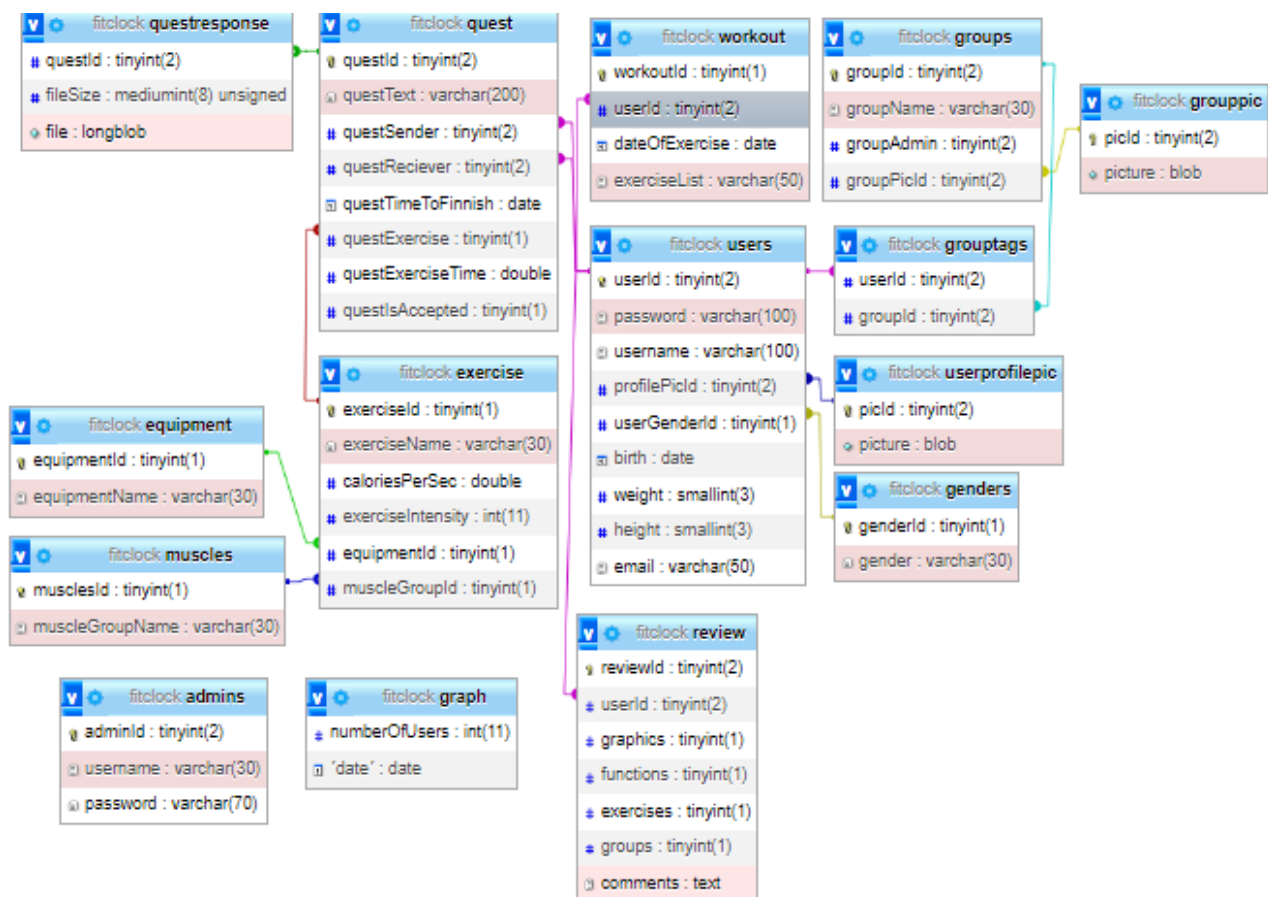
Webszerver verziója: 2.4.39

Adatbázis neve: fitclock

Tárolómotor: InnoDB

Alapértelmezett illesztés: utf8_hungarian_ci

1.4.1 Táblák kapcsolata



1.4.2 A táblák

1.4.2.1 A `users` tábla

A felhasználó személyes adatait tárolja, melyeket regisztrációnál kell megadni. Bejelentkezéshez ebből a táblából olvassa ki az adatokat. A felhasználó regisztráláskor adja meg a szükséges adatokat, melyet később is tud frissíteni, változtatni. A biztonságos adattárolás érdekében a felhasználó jelszavát titkosítva tároljuk. Az email-címet validáltatjuk mielőtt elfogadnánk a regisztrációt, így biztosítva, hogy valós email-címet használjanak, így a későbbiekben kaphatnak e-mailes értesítést is.

Szerkezete:

Oszlop	Adattípus	Leírás	Kulcs
userId	egész szám	Felhasználó azonosítója	PK
password	szöveg(100)	Felhasználó jelszava	
username	szöveg(100)	Felhasználó teljes neve	
profilePicId	egész szám	A felhasználó profilképe (<i>userProfilePic</i> táblából)	FK
userGenderId	egész szám	A felhasználó neme (<i>genders</i> táblából)	FK
birth	dátum	Felhasználó születési dátuma	
height	egész szám	Felhasználó magassága centiméterben	
weight	egész szám	Felhasználó testsúlya kilogrammban	
email	szöveg(50)	Felhasználó email címe	

Létrehozása:

```
CREATE TABLE IF NOT EXISTS fitclock.users(  
    userId TINYINT(2) NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    password VARCHAR(100) NOT NULL,  
    username VARCHAR(100) NOT NULL,  
    profilePicId TINYINT(2) NOT NULL DEFAULT '1',  
    userGenderId TINYINT(1) NOT NULL,  
    birth DATE,  
    weight SMALLINT(3),  
    height SMALLINT(3),  
    email VARCHAR(50) NOT NULL,  
    FOREIGN KEY (profilePicId) REFERENCES userProfilePic(picId),  
    FOREIGN KEY (userGenderId) REFERENCES genders(genderId)  
) ENGINE=InnoDB;
```

1.4.2.2 A `genders` tábla

A nemeket tartalmazza a felhasználók táblához. Adatait az adatbázis létrehozása után hozzárendeljük.

Szerkezete:

Oszlop	Adattípus	Leírás	Kulcs
genderId	egész szám	Azonosító	PK
gender	szöveg(30)	Nem szövegesen	

Létrehozása:

```
CREATE TABLE IF NOT EXISTS fitclock.genders(  
    genderId TINYINT(1) NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    gender VARCHAR(30)  
) ENGINE=InnoDB;
```

1.4.2.3 A `userProfilePic` tábla

A felhasználók profilképét tárolja, az első rekord tartalmazza az alapértelmezett képet.

Szerkezete:

Oszlop	Adattípus	Leírás	Kulcs
picId	egész szám	Azonosító	PK
picture	fájl	A felhasználó profilképe	

Létrehozása:

```
CREATE TABLE IF NOT EXISTS fitclock.userProfilePic(  
    picId TINYINT(2) NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    picture BLOB  
) ENGINE=InnoDB;
```

1.4.2.4 Az `equipment` tábla

Az edzésekhez szükséges eszközöket tárolja.

Szerkezete:

Oszlop	Adattípus	Leírás	Kulcs
equipmentId	egész szám	Eszköz azonosítója	PK
equipmentName	szöveg(30)	Eszköz neve	

Létrehozása:

```
CREATE TABLE IF NOT EXISTS fitclock.equipment(  
    equipmentId TINYINT(1) NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    equipmentName VARCHAR(30) NOT NULL  
) ENGINE=InnoDB;
```

1.4.2.5 Az `exercise` tábla

Tartalmazza a gyakorlatokat, és az azokhoz tartozó adatokat. Kalória alapértelmezetten egy percre utal.

Szerkezete:

Oszlop	Adattípus	Leírás	Kulcs
exerciseld	egész szám	A gyakorlat azonosítója	PK
exerciseName	szöveg(30)	Gyakorlat neve	
CaloriesPerSec	lebegőpontos szám(double)	Egy másodperc alatt elégetett kalóriák száma	
exerciseIntensity	egész szám	A gyakorlat inenzitása	
equipmentId	egész szám	A gyakorlathoz használt felszerelése azonosítója (<i>equipment</i> táblából)	FK
muscleGroupList	varchar(50)	a gyakorlat során használt izomcsoportok azonosítóinak listája	

Létrehozása:

```
CREATE TABLE IF NOT EXISTS fitclock.exercise(  
    exerciseId TINYINT(1) NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    exerciseName VARCHAR(30) NOT NULL,  
    caloriesPerSec DOUBLE,  
    exerciseIntensity INT,  
    equipmentId TINYINT(1),  
    muscleGroupList VARCHAR(50) NOT NULL,  
    FOREIGN KEY (equipmentId) REFERENCES equipment(equipmentId),  
    FOREIGN KEY (muscleGroupId) REFERENCES muscles(musclesId)  
) ENGINE=InnoDB;
```

1.4.2.6 A `muscles` tábla

Az izomcsoportokat tárolja el.

Szerkezete:

Oszlop	Adattípus	Leírás	Kulcs
muscleId	egész szám	Az izomcsoport azonosítója	PK
muscleGroupName	szöveg(30)	Az izomcsoport neve	

Létrehozása:

```
CREATE TABLE IF NOT EXISTS fitclock.muscles(  
    musclesId TINYINT(1) NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    muscleGroupName VARCHAR(30) NOT NULL  
)ENGINE=InnoDB;
```

1.4.2.7 A `workout` tábla

A gyakorlat adatait foglalja magában. Több gyakorlatot egy edzés során külön rekordokba rögzíti. Az idő az alkalmazásban kerül átváltásra. Az egy edzés során végzett gyakorlatokat a gyakorlat azonosítója alapján tudjuk lekérdezni.

Szerkezete:

Oszlop	Adattípus	Leírás	Kulcs
workoutId	egész szám	Edzés azonosítója	PK
userId	egész szám	Felhasználó azonosítója (<i>users</i> táblából)	FK
dateOfExercise	dátum	Edzés időpontja	
exerciseList	varchar(50)	Gyakorlat azonosítója (<i>exercise</i> táblából)	FK
exerciseTimeList	varchar(50)	Gyakorlatok idejei	

Létrehozása:

```
CREATE TABLE IF NOT EXISTS fitclock.workout(  
    workoutId TINYINT(1) NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    userId TINYINT(2) NOT NULL,  
    dateOfExercise DATE,  
    exerciseList VARCHAR(50) NOT NULL,  
    exerciseTimeList VARCHAR(50) NOT NULL,  
    FOREIGN KEY (userId) REFERENCES users(userId)  
)ENGINE=InnoDB;
```

1.4.2.8 A `groupPic` tábla

A csoportok profilképét tárolja, az első rekord az alapértelmezett képet tartalmazza.

Szerkezete:

Oszlop	Adattípus	Leírás	Kulcs
picId	egész szám	Azonosító	PK
picture	fájl	A képfájl	

Létrehozása:

```
CREATE TABLE IF NOT EXISTS fitclock.groupPic(  
    picId TINYINT(2) NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    picture BLOB  
) ENGINE=InnoDB;
```

1.4.2.9 A `groupTags` tábla

A felhasználók azonosítóját rendeli csoportazonosítóhoz.

Szerkezete:

Oszlop	Adattípus	Leírás	Kulcs
userId	egész szám	A felhasználó azonosítója (<i>users</i> táblából)	FK
groupId	egész szám	A csoport azonosítója (<i>groups</i> táblából)	FK

Létrehozása:

```
CREATE TABLE IF NOT EXISTS fitclock.grouptags(  
    userId TINYINT(2) NOT NULL,  
    groupId TINYINT(2) NOT NULL,  
    FOREIGN KEY (userId) REFERENCES users(userId),  
    FOREIGN KEY (groupId) REFERENCES groups(groupId)  
) ENGINE=InnoDB;
```


1.4.2.10 A `groups` tábla

A csoportokat hozza létre. Azonosítja a csoport adminját és a csoport nevét. Egy felhasználó csak egy csoporthoz tartozhat.

Szerkezete:

Oszlop	Adattípus	Leírás	Kulcs
groupId	egész szám	Csoport azonosítója	PK
groupName	szöveg(30)	A csoport neve	
groupAdmin	egész szám	Csoport adminjának azonosítója	
groupPicId	egész szám	A csoport képének azonosítója	

Létrehozása:

```
CREATE TABLE IF NOT EXISTS fitclock.groups(  
    groupId TINYINT(2) NOT NULL AUTO_INCREMENT PRIMARY KEY,  
    groupName VARCHAR(30) NOT NULL,  
    groupAdmin TINYINT(2) NOT NULL,  
    groupPicId TINYINT(2),  
    FOREIGN KEY (groupPicId) REFERENCES groupPic(picId)  
) ENGINE=InnoDB;
```

1.4.2.11 A `questResponse` tábla

A videó választ tartalmazza. A fájl méret a fájl ellenőrzéséhez szükséges.

Szerkezete:

Oszlop	Adattípus	Leírás	Kulcs
responseld	egész szám	Válasz azonosítója	PK
questId	egész szám	Kihívás azonosítója (quests táblából)	FK
fileSize	egész szám	Fájl mérete	
file	longblob	Videó válasz	

Létrehozása:

```
CREATE TABLE IF NOT EXISTS fitclock.questResponse(  
    questId TINYINT(2) NOT NULL,  
    fileSize MEDIUMINT UNSIGNED NOT NULL,  
    file LONGBLOB NOT NULL,  
    FOREIGN KEY (questId) REFERENCES quest(questId)  
) ENGINE=InnoDB;
```

1.4.2.12 A `quest` tábla

A kihívásra vonatkozó adatokat tartalmazza, illetve azt, hogy el lett e fogadva, valamint a választ videó formátumban.

Szerkezete:

Oszlop	Adattípus	Leírás	Kulcs
questId	egész szám	Kihívás azonosítója	PK
questText	szöveg(200)	Kihívás üzenete	
questSender	egész szám	Kihívás küldőjének azonosítója (users táblából)	FK
questReciever	egész szám	Kihívott azonosítója (users táblából)	FK
questWorkout	egész szám	Kihívás gyakorlat azonosítója (exercise táblából)	FK
questExerciseTime	lebegőpontos szám(double)	Gyakorlat ideje	
questIsAccepted	boolean	Kihívás el lett e fogadva	

Létrehozása:

```
CREATE TABLE IF NOT EXISTS fitclock.quest(  
    questId TINYINT(2) NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    questText VARCHAR(200),  
    questSender TINYINT(2) NOT NULL,  
    questReciever TINYINT(2) NOT NULL,  
    questExercise TINYINT(1) NOT NULL,  
    questExerciseTime DOUBLE NOT NULL,  
    questIsAccepted BOOLEAN NOT NULL,  
    FOREIGN KEY (questSender) REFERENCES users(userId),  
    FOREIGN KEY (questReciever) REFERENCES users(userId),  
    FOREIGN KEY (questExercise) REFERENCES exercise(exerciseId)  
) ENGINE=InnoDB;
```

1.4.2.13 A `graph` tábla

Az asztali alkalmazás grafikonjához tartalmazza az adatokat.

Szerkezete:

Oszlop	Adattípus	Leírás	Kulcs
numberOfUsers	egész szám	A felhasználók száma	
date	dátum	Mérés időpontja	

Létrehozása:

```
CREATE TABLE IF NOT EXISTS fitclock.graph(  
    numberOfUsers INT NOT NULL,  
    `date` DATE  
) ENGINE=InnoDB;
```

1.4.2.14 A `review` tábla

A mobil appról és a webről beérkező értékelési adatokat tárolja, melyet az asztali alkalmazásban dolgozunk fel.

Szerkezete:

Oszlop	Adattípus	Leírás	Kulcs
reviewId	egész szám	Azonosító	PK
userId	egész szám	A felhasználó azonosítója	
graphics	egész szám	Grafikára adott pontok ötös skálán	
functions	egész szám	Funkciókra adott pontok ötös skálán	
exercises	egész szám	Feladatokra adott pontok ötös skálán	
groups	egész szám	Csoportok funkcióira adott pontok ötös skálán	
comment	szöveg	Megjegyzés, szöveges értékelés	

Létrehozása:

```
CREATE TABLE IF NOT EXISTS fitclock.review(  
    reviewId TINYINT(2) NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    userId TINYINT(2) NOT NULL,  
    graphics TINYINT(1) NOT NULL,  
    functions TINYINT(1) NOT NULL,  
    exercises TINYINT(1) NOT NULL,  
    groups TINYINT(1) NOT NULL,  
    comments TEXT,  
    FOREIGN KEY (userId) REFERENCES users(userId)  
) ENGINE=InnoDB;
```

1.4.2.15 Az `admins` tábla

Az asztali alkalmazáshoz tárolja a bejelentkezési adatokat.

Szerkezete:

Oszlop	Adattípus	Leírás	Kulcs
adminId	egész szám	Azonosító	PK
username	szöveg(30)	Az admin felhasználó neve	
password	szöveg(70)	Az admin jelszava titkosítva	

Létrehozása:

```
CREATE TABLE IF NOT EXISTS fitclock.admins(  
    adminId TINYINT(2) NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    username VARCHAR(30) NOT NULL,  
    password CHAR(70) NOT NULL  
)ENGINE=InnoDB;
```

1.5 Jogosultságok

1.5.1 Feladatkörök

	Nem bejelentkezett felhasználó	Bejelentkezett felhasználó	Csoporttag	Csoport admin	Admin
Egyéni edzésterv mentése	✓	✓	✓	✓	
Edzések statisztikái		✓	✓	✓	
Ajánlott gyakorlatok		✓	✓	✓	
Csoporthoz való csatlakozás		✓	✓		
Összes csoporttag kihívása				✓	
Csoporttag kihívás			✓	✓	
Alkalmazásra vonatkozó statisztikák					✓
Értékelések felügyelete					✓
Értékelés küldése	✓	✓	✓	✓	
Statisztikák a felhasználókkal kapcsolatban				✓	✓

1.5.2 Jogosultsági szintek

1.szint	Admin		
2.szint	Csoport admin		
3.szint	Csoporttag	Bejelentkezett felhasználó	
4.szint			Nem bejelentkezett felhasználó

1.6 Az asztali alkalmazás felépítése

Az asztali alkalmazás Windows Forms Application(.NET Framework)-ben készült az objektum orientált programozás elve és a tiszta kód elve szerint C# programnyelven. Az MVVM modell alapján lett felépítve a szerkezete.

Az asztali alkalmazást valamint annak tesztjeit Borka készítette el.

1.6.1 Model

Az adatbázishoz csatlakozáshoz való parancsokat tartalmazza, valamint a meghívható függvényeket, amelyek különböző értékekkel térhetnek vissza.

Itt található a szerverhez való kapcsolódások, valamint a lekéréseket végrehajtó függvényeket tartalmazó osztály.

1.6.2 ViewModel

A Model-ben található függvények visszatérési értékeit hívja meg. A lekérési parancsot a függvény paramétereiként küldi át. A visszakapott értékeket függvényekben térnek vissza az elvégzett számítások után.

Itt találhatóak az admin validációt, a becsléseket, valamint az eseményeket tartalmazó osztályok.

1.6.3 View

A ViewModel-ben feldolgozott értékeknek az eredményeit megjeleníti, valamint a felületre vonatkozó stílus beállításokat tartalmazza.

Itt találhatóak a felület kinézetére vonatkozó osztályok, valamint a Form elemek.

1.6.4 A felületek

1.6.4.1 A Login felület

A felületen a már regisztrált adminok jelentkezhetnek be. A parancsok a gomb lenyomására futnak le. A jelszót titkosítva olvassa be az adatbázisból, ezért a jelszót titkosítja az összehasonlításához. Amennyiben a validáció sikeres belépteti az admint. A regisztrálás felirat a regisztrációs felülethez továbbít.

1.6.4.2 A Registration felület

Az új adminok regisztrálására szolgál. A sikeres regisztrációhoz szükséges a hitelesítő kód, különben nem tölti fel az adatokat az adatbázisba. A jelszót sha256-tal titkosítja feltöltés előtt.

Az admin létrehozása előtt megvizsgálja, hogy létezik e ugyanolyan a felhasználónév-jelszó pár már az adatbázisban.

1.6.4.3 A Dashboard felület

A ViewModel-ből bekéri a kiszámított átlagokat a 4 kategóriára, meghívja a diagramba a kiszámolt adatokat, valamint a növekedés mértékét a legutóbbi mérés óta.

1.6.4.4 A Comment felület

A szöveges értékeléseket Deep Learninggel kategorizálva jeleníti meg. A kategorizálás a ViewModel predictions mappájában lévő osztályokban történik. A Data mappa tartalmazza a betanításhoz szükséges adatokat egy .txt fájlban. A tanítás után az adatokat az adatbázisból feldolgozza és eltárolja. A nyelvi beállítások miatt csak a legalább 35%-ig pontosan kategorizált adatokat jeleníti meg a felületen.

1.6.5 A tanítás menete

A tanításhoz szükséges .txt fájl következőképpen épül fel:

```
Szuper. 1
Nem jó . 0
Nagyon rossz. 0
Imádtam. 1
Kiváló. 1
okés 1
Mérges vagyok és vissza akarom kapni a pénzem. 0
Nem éreztem annyira jónak. 0
Remek élmény. 1
Kitűnő. 1
Nem tölteném le újra. 0
It was just bad. 0
I loved it. 1
```

A megfelelő pontosság érdekében szükséges angol nyelvű szöveget is megadni a betanításhoz.

```
ITransformer BuildAndTrainModel(MLContext mlContext, IDataView splitTrainSet)
{
    var estimator = mlContext.Transforms.Text
        .FeaturizeText(outputColumnName: "Features",
            inputColumnName: nameof(SentimentData.SentimentText))
        .Append(mlContext.BinaryClassification.Trainers
            .SdcaLogisticRegression(labelColumnName: "Label", featureColumnName: "Features"));
    var model = estimator.Fit(splitTrainSet);
    return model;
}
```

A szöveg mögött található szám jelzi, hogy az adott komment pozitív vagy negatív.

A függvény két paramétert kap meg, az első a Microsoft.ML egyik osztálya melynek meghívása szükséges az elvégzett műveletek végrehajtásához. A második a betöltött szöveges fájlt a szükséges trainsetbe splitelve adja át.

A Features oszlopként hivatkozott oszlop tartalmazza komment szövegét, míg a Label oszlop tartalmazza a bináris számot.

Az estimator változóban megformázzuk, hogy milyen módon vannak az adatok, illetve az oszlopokból megtanulja, milyen jellegű szöveghez milyen label tartozik, ezzel felépítettük a modellünket.

A model változóban ezután lefuttatjuk a training-et. A végén a feldolgozott adatokat küldi vissza a függvény.¹²

¹² <https://learn.microsoft.com/en-us/dotnet/machine-learning/tutorials/sentiment-analysis-cli>

1.6.6 A kerekített ablakok

```
class RoundedRectangle
{
    [DllImport("Gdi32.dll", EntryPoint = "CreateRoundRectRgn")]
    18 references
    public static extern IntPtr CreateRoundRectRgn
    (
        int nLeftRect,
        int nTopRect,
        int nRightRect,
        int nBottomRect,
        int nWidthEllipse,
        int nHeightEllipse
    );
}
```

Az ablakok kerekítésére szolgáló osztály Borka régebbi projektjéből származik.

A gdi32.dll a Windows GDI(Graphical Device Interface)-nak tartalmaz funkciókat, amely így lehetővé teszi egyszerű 2 dimenziós objektumok használatát.

A CreateRoundRectRgn a Windows API egyik osztálya, amely egy téglalap alakú, lekerekített sarkú régiót hoz létre. a form beállításainál a régiónál tudjuk meghívni az osztályt a megadott

```
loginForm.Region = Region.FromHrgn(RoundedRect.CreateRoundRectRgn(0, 0, 300, 450, 25, 25));
```

értékekkel.

1.6.7 A regisztráció menete

A regisztrációhoz egy külső osztály kerül meghívásra. Az osztály teszteli, hogy ne legyen ugyanolyan nevű felhasználó az adatbázisban, valamint az adatbázis védelme érdekében, hogy ne legyen könnyen feltörhető jelszó.

```
public void Register(string username, string password, string key)
{
    if (PasswordSafe(password))
    {
        string keyForAuthorization = "123Uc!22vy";
        AdminValiditation av = new AdminValiditation(username, password);
        if (key == keyForAuthorization)
        {
            if (av.Validation() == "felhasznalo hiba")
            {
                CreateAdmin(username, Sha256(password));
                login.ShowDialog();
            }
        }
        else
        {
            el.Show();
            el.errorTypeLb.Text = "Nem megfelelő hitelesítési kulcs!!";
            el.errorLb.Text = "HIBA A REGISZTRÁCIÓ SORÁN";
        }
    }
    else
    {
        r.ShowDialog();
    }
}
```

1.7 A mobil alkalmazás felépítése

1.7.1 Images

Az alkalmazás menüpontjainak ikonjait tartalmazó mappa.

1.7.2 MarkupExtensions

Az XAML kódhoz használt egyedi készítésű kiegészítés, aminek segítségével URL helyett a lokális fájlokhoz tudjuk csatolni a képek elérési útvonalát.

1.7.3 Model

Itt helyezkednek el a menüpontok szerkezetét tartalmazó osztályok, és az alkalmazásban használt komplexebb objektumok, például a gyakorlatok, edzések szerkezete.

1.7.4 Renderer

Egyedi készítésű renderer, hogy a szövegbeviteli mezők színe illeszkedjen az alkalmazásban használt színpalettába.

1.7.5 Services

Itt helyezkedik el az adatbázis elkészüléséig használt MockdataStore, és az ehhez tartozó interface. Az SQLite adatbázishoz tartozó interface. A képernyő alján megjelenő „Toast” felugró üzenet implementálásához használt interface.

1.7.6 Felületek

Az alkalmazás felületeit alkotó `ContentPage`-ek, és a hozzájuk tartozó „Code Behind” osztályok.

App.xaml - App.xaml.cs

Az XAML file az alkalmazás szintű stílus szabályokat tartalmazza.

A Code Behind pedig az app belépő pontja, itt adjuk meg az első tényleges felületet `MainPage` property-ként. Ez a `LoginPage`.

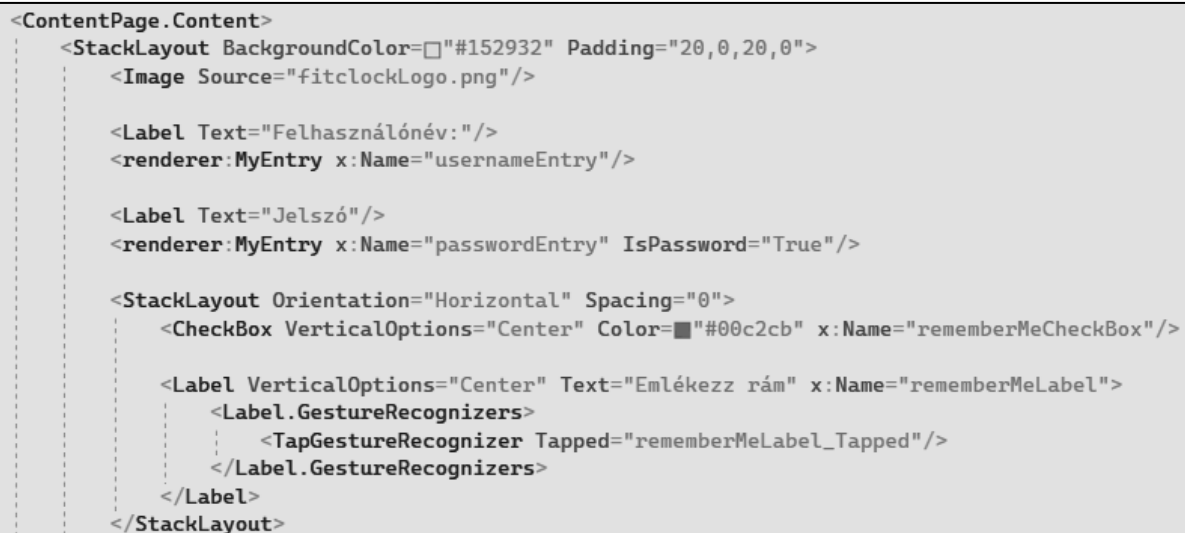


```
namespace FitClock
{
    6 references
    public partial class App : Application
    {
        1 reference
        public App()
        {
            InitializeComponent();

            MainPage = new NavigationPage(new LoginPage());
        }
    }
}
```

LoginPage.xaml - LoginPage.xaml.cs

Az XAML file egy `ContentPage`, amin belül egy `StackLayout` rendezi az elemeket egy függőleges sorba, feliratok és egyedi szövegbeviteli mezők egymás után a felhasználónév és jelszó beviteléhez. Az emlékezz rám felirathoz `TapGestureRecognizer` van hozzárendelve, így arra koppintva is változik a checkbox állapota.



```
<ContentPage.Content>
    <StackLayout BackgroundColor="#152932" Padding="20,0,20,0">
        <Image Source="fitclockLogo.png"/>

        <Label Text="Felhasználónév:"/>
        <renderer:MyEntry x:Name="usernameEntry"/>

        <Label Text="Jelszó"/>
        <renderer:MyEntry x:Name="passwordEntry" IsPassword="True"/>

        <StackLayout Orientation="Horizontal" Spacing="0">
            <CheckBox VerticalOptions="Center" Color="#00c2cb" x:Name="rememberMeCheckBox"/>

            <Label VerticalOptions="Center" Text="Emlékezz rám" x:Name="rememberMeLabel">
                <Label.GestureRecognizers>
                    <TapGestureRecognizer Tapped="rememberMeLabel_Tapped"/>
                </Label.GestureRecognizers>
            </Label>
        </StackLayout>
    </ContentPage.Content>
```

A CodeBehind ellenőrzi, hogy van-e elmentve az felhasználónév és jelszó, ezeket magától kitölti. A bejelentkezés gombra kattintva egy felhasználó objektum listába lekérjük az felhasználókat a MockDataStore-ból. Egy új felhasználót hozunk létre, amit egy Linq kifejezéssel az előbbi listából keresünk. Keressük azt a felhasználót, akinek a felhasználóneve megegyezik a bevitt felhasználónév értékkel. A linq kifejezést egy try-catch szerkezetbe ágyazzuk, így ha az nem talál megfelelő felhasználót, akkor a catch részbe írt kód fog lefutni. Létező felhasználónév esetén egy BCrypt ellenőrzés dönti el, hogy megfelelő jelszó párosul-e a felhasználónévhez.

```
private async void LoginButton_Clicked(object sender, EventArgs e)
{
    List<User> alluser = await Database.GetUsers();

    try
    {
        User theUser = alluser.First(x => x.username == usernameEntry.Text);

        if (BCrypt.Net.BCrypt.Verify(passwordEntry.Text, theUser.password))
        {
            saveData();
            Application.Current.Properties["userId"] = theUser.userId;
            Application.Current.MainPage = new NavigationPage(new MainPage());
        }
        else
        {
            await DisplayAlert("Sikertelen bejelentkezés!", "Hibás jelszót adott meg, próbálja újra!", "OK");
            passwordEntry.Text = String.Empty;
        }
    }
    catch
    {
        await DisplayAlert("Sikertelen bejelentkezés!", "Nem létezik a felhasználói fiók, regisztráljon új fiókot!", "OK");
    }
}
```

FlyoutMenu.xaml

A navigációs menü szerkezete, és kinézete is itt van létrehozva, továbbá a FlyoutMenuItem model alapján, a menüpontok is itt vannak létrehozva.

```
<ListView x:Name="FlyoutMenuListView" x:FieldModifier="public">
    <ListView.ItemsSource>
        <x:Array Type="{x:Type models:FlyoutMenuItem}">
            <models:FlyoutMenuItem Title="Edzőstervek" IconSource="{custom:EmbeddedImage ResourceId=FitClock.Images.dumbbell.png}" TargetPage="{x:Type local:MyWorkoutsPage }"/>
            <models:FlyoutMenuItem Title="Új edzősterv készítése" IconSource="{custom:EmbeddedImage ResourceId=FitClock.Images.create.png}" TargetPage="{x:Type local:CreateWorkoutPage}"/>
            <models:FlyoutMenuItem Title="Értékelj az appot" IconSource="{custom:EmbeddedImage ResourceId=FitClock.Images.rate.png}" TargetPage="{x:Type local:RateThisAppPage}"/>
            <models:FlyoutMenuItem Title="Beállítások" IconSource="{custom:EmbeddedImage ResourceId=FitClock.Images.settings.png}" TargetPage="{x:Type local:SettingsPage}"/>
            <models:FlyoutMenuItem Title="Kijelentkezés" IconSource="{custom:EmbeddedImage ResourceId=FitClock.Images.logout.png}" TargetPage="{x:Type local:LoginPage}"/>
        </x:Array>
    </ListView.ItemsSource>
    <ListView.ItemTemplate>
        <DataTemplate>
            <ViewCell>
                <Grid>
                    <Grid.ColumnDefinitions>
                        <ColumnDefinition Width="33"/>
                        <ColumnDefinition Width="*" />
                    </Grid.ColumnDefinitions>
                    <Image Margin="5" Source="{Binding IconSource}"/>
                    <Label VerticalOptions="Center" Grid.Column="1" Text="{Binding Title}"/>
                </Grid>
            </ViewCell>
        </DataTemplate>
    </ListView.ItemTemplate>
</ListView>
```

MainPage.xaml - MainPage.xaml.cs

A navigációs menü koppintása az ehhez tartozó CodeBehind-ban van kezelve. A kilépésnél egy felugró ablak megerősítést kér a felhasználtól.

```
private async void FlyoutMenuListView_ItemTapped(object sender, ItemTappedEventArgs e)
{
    var item = e.Item as FlyoutMenuItem;
    if (item != null)
    {
        if (item.TargetPage == typeof(LoginPage))
        {
            if (!await DisplayAlert("Kilépés", "Biztosan ki akar lépni?", "Igen", "Nem"))
            {
                item.TargetPage = typeof(MainPage);
            }
        }

        Detail = new NavigationPage((Page)Activator.CreateInstance(item.TargetPage));
        FlyoutMenuFlyout.FlyoutMenuListView.SelectedItem = null;
        IsPresented = false;
    }
}
```

MyWorkoutsPage.xaml - MyWorkoutsPage.xaml.cs

A CodeBehind lekéri az SQLite adatbázisból az edzésterveket, majd egy ObservableCollection segítségével a felülethez csatolja azokat.

```
//Current user ID
private int _userId = Convert.ToInt32(App.Current.Properties["userId"]);
//Connection to local db
private SQLiteAsyncConnection _connection;

//Source for listview
private ObservableCollection<Workout> _workouts;

1 reference
public MyWorkoutsPage()
{
    InitializeComponent();
    _connection = DependencyService.Get<ISQLiteDb>().GetConnection();

    myWorkoutsListView.ItemTapped += myWorkoutsListViewItem_Tapped;
}

0 references
protected override async void OnAppearing()
{
    //Create workout table if not exists
    await _connection.CreateTableAsync<Workout>();
    //Fill observable collection with data from table
    var allWorkout = new ObservableCollection<Workout>(await _connection.Table<Workout>().ToListAsync());

    _workouts = new ObservableCollection<Workout>(allWorkout.Where(x => x.userId == _userId));
    //Bind listview to observable collection
    myWorkoutsListView.ItemsSource = _workouts;

    base.OnAppearing();
}
```

RateThisAppPage.xaml.cs

A felhasználónként egyszeri értékelés ellenőrzése egy SQLite adatbázissal történik.

```
private int _userId = Convert.ToInt32(Application.Current.Properties["userId"]);
private SQLiteAsyncConnection _connection;
0 references
public RateThisAppPage()
{
    InitializeComponent();
    _connection = DependencyService.Get<ISQLiteDb>().GetConnection();

    isAvailable();
}
2 references
private async void isAvailable()
{
    string query = "SELECT COUNT(*) FROM Rating WHERE userId="+userId+";";
    int count = await _connection.ExecuteScalarAsync<int>(query);
    if (count != 0)
    {
        ratingButton.IsEnabled = false;
        ratingButton.Text = "Értékelés elküldve!";
    }
}
```

1.8 A weboldal felépítése

A weboldal REST API-ban készült, Twig template-et használva és Composer függőségkezelőt használva. A weboldal az objektum orientált programozás és a tiszta kód elve szerint készült PHP, HTML és Javascript nyelveken.

Az MVC modell alapján lett felépítve a szerkezete.

A teljes weboldalt és annak tesztelését Lehel készítette el, full stack programozóként.

A weboldal front-end része teljesen egyedi, Lehel által lett készítve, a megbeszél stílus, és színskála alapján.

1.8.1 App

A weboldalhoz szükséges mappákat tárolja.

1.8.1.1 Controller

A web logikai kéréseit és műveleteit kezeli. Legtöbbször a Model-nek ad át és kér tőlük adatokat, majd ezekkel az adatokkal végez műveleteket. A felhasználói felületet is itt generáljuk le a View-ban található Twig template-ekkel.

1.8.1.2 css

A felhasználói felületre vonatkozó stíluslapokat tartalmazza. Egyedileg formázza meg a View-ban létrehozott twig template-eket. Javascript segítségével együtt a weboldal reszponzivitását is segítik.

1.8.1.3 img

A weboldalon használt állandó képeket tartalmazza. Ezek nem azonosak az adatbázison eltárolt képekkel.

1.8.1.4 js

A megfelelő megjelenítéshez, reszponzivitáshoz, valamint adatfeldolgozáshoz szükséges szkripteket tartalmazza. A javascript kezeli a diagramok létrehozását is.

1.8.1.5 Lib

A szerver kapcsolathoz és válaszokhoz szükséges osztályokat tartalmazza.

1.8.1.6 Model

Az adattáblákat kezelő osztályokat tartalmazza. Kommunikál az adatbázissal, adatokat kér le, vagy tölt fel oda. A Controller számára fontos kérdéseket is eldönthet, mint például, hogy az adott email címmel regisztráltak-e már felhasználót vagy sem. A jelszó titkosítás is itt történik, közvetlenül az adat feltöltése előtt.

1.8.1.7 View

Az oldalon megjelenített felhasználói felületeket tartalmazza.

1.8.2 Vendor

a „composer.json” fájl alapján generálódik le. A weboldalhoz szükséges egyéb technológiákat tartalmazza.

1.8.3 index.php

A felhasználó az index.php-n keresztül kommunikál a weboldal többi részével. Itt találhatóak a belépési pontok, amiken keresztül egyes funkciók elérhetők, mint például egy twig template legenerálása. Itt található a funkció, amely elindítja a SESSION-t és eldönti, hogy egy felhasználó már be van-e jelentkezve a weboldalra. Csak a már bejelentkezett felhasználók kommunikálhatnak ezáltal a weboldal belső elemeivel.

1.8.4 newpic.jpg és newProfPic.jpg

A csapatok és a felhasználók profil kép feltöltését elősegítő, nem állandó kép. A legutolsó felhasználó által feltöltött profil kép tömörített verzióját tartalmazza, amely az adatbázisba kerül feltöltésre utána.

1.8.5 A felületek

1.8.5.1 Login felület

A weblap megnyitásával, ide irányítja a felhasználót az index.php fájl. Itt a már regisztrált felhasználók jelentkezhetnek be. A LoginController login() funkciója először lekérdezi a loginDAO getUser() funkció segítségével, hogy az adatbázisban található-e a bejelentkezéskor megadott adatok szerinti felhasználó. A loginDAO getUser() funkciója számításba veszi a titkosítást is a felhasználó keresése során. Amennyiben létezik már a felhasználó, elindít egy SESSION-t és eltárolja a fontosabb felhasználói adatokat vele, majd megjeleníti a userStatistics.html.twig oldalt, ezáltal beléptetve a felhasználót. Ellenkező esetben, hibás adatok esetén, értesíti a felhasználót, és újratölti a login felületet.

```
public function login(){
    $userEmail= $_POST['userEmail'];
    $password=$_POST['password'];

    $userData = loginDAO::getUser($userEmail, $password);
    $twig = (new LoginController())->setTwigEnvironment();

    if($userData!=NULL){
        $id=$userData->userId;
        $picId=$userData->profilePicId;
        $userName=$userData->username;

        session_start();
        $_SESSION['userId']=$id;
        $_SESSION['userName']=$userName;
        $_SESSION['profPicId']=$picId;

        $userProfilePicture = loginDAO::getProfilePicture($picId);

        echo $twig->render('statistics/userStatistics.html.twig', ['userData'=>$userData, 'profPic'=>$userProfilePicture]);
    }
    else{
        echo '<script>alert("Hibás felhasználónév vagy jelszó.")</script>';
        header("Refresh:0; url=/showlogin");
    }
}
```

1.8.5.2 User statistics felület

Sikeres bejelentkezés után ide van átirányítva a felhasználó. A statisticsController-ben lévő funkciók, kezelik a statisticsDAO-ban lekért adatokat, és a szükséges, akár átalakított adatokat továbbítja a userStatistics.html.twig template-nek, amelyekkel a userStatistics javascript funkciói, a twig template-n lévő gombokra való kattintás esetén létrehoznak egy diagramot a megfelelő adatokból.

1.8.5.3 Registration felület

A login felületen található regisztráció link, a regisztrációs felületre viszi a felhasználót.

```
public function registration(){
... $userEmail=$_POST['userEmail'];
... $userName=$_POST['userName'];
... $userPassword=$_POST['userPassword'];
... $userPasswordSecond=$_POST['userPasswordSecond'];
... $userBirthDate=$_POST['userBirthDate'];
... $userSex=$_POST['userSex'];
...
... if(loginDAO::userIsRegistered($userEmail)){
...     echo '<script>alert("Ezzel az emaillel már regisztrált felhasználó.")</script>';
...     header("Refresh:0; url=/showlogin");
... }else{
...     if($userPassword!=$userPasswordSecond){
...         echo '<script>alert("A jelszavak nem egyeznek.")</script>';
...         header("Refresh:0; url=/showRegister");
...     }
...     else{
...         loginDAO::createUser($userEmail, $userName, $userPassword, $userBirthDate, $userSex);
...         header("Refresh:0; url=/showlogin");
...         echo '<script>alert("Sikeres regisztráció.")</script>';
...     }
... }
}
```

Majd az adatok megadása után, a gombra kattintva a LoginController registration() funkciója elkezd a regisztrációt, amennyiben a regisztrációs oldalon a jelszó megfelelt a megadott követelményeknek, a követelmények, hogy legalább 6 karakterből álljon, legalább 1 nagybetű és 1 szám szerepeljen benne. A regisztráció előtt a registration() funkció ellenőrzi, hogy a megadott email-címmel regisztrált-e már másik felhasználó. Amennyiben igen, figyelmezteti a felhasználót, hogy az email címmel már regisztráltak, amennyiben még nem és a két különböző „jelszó” mezőbe beírt jelszavak egyeznek elvégzi a regisztrációt loginDAO createuser() funkciójával együtt, ahol a jelszó titkosításra is kerül. A titkosítást BCrypt-ben végeztük, mivel kutatásaink alapján, nem csak biztonságos, de időtálló is.

```
public static function createUser(string $userEmail, string $userName, string $userPassword, string $userBirthDate, int $userSex){
... $dbObj = new Database();
... $conn = $dbObj->getConnection();
... $encryptedPass=password_hash($userPassword, PASSWORD_BCRYPT);
...
... $statement=$conn->prepare("INSERT INTO users (email, username, password, birth, userGenderId)
... VALUES (:userEmail,:userName,:userPassword,:userBirthDate,:userSex);");
...
... $statement->execute([
...     'userEmail'=>$userEmail,
...     'userName'=>$userName,
...     'userPassword'=>$encryptedPass,
...     'userBirthDate'=>$userBirthDate,
...     'userSex'=>$userSex,
... ]);
}
```

1.8.5.4 Profile settings felület

A menüszalagon a „Profil beállítások” kiválasztása esetén, ide irányítja át a felhasználót. Ezen az oldalon, a felhasználónak az adatai láthatók, amiért a profileDAO getProfileData() funkciója felelős. Ez lekéri a bejelentkezett felhasználó ID-ja alapján a többi adatát, majd visszaküldi a profileController showProfile() funkciójának, hogy az oldal betöltésekor meg tudja jeleníteni.

Amennyiben a felhasználó változtatni szeretne az adatokon és rákattint a „Mentés” gombra, a profileController setProfile() funkciója elvégzi a megfelelő feltételek alapján, a profileDAO megfelelő funkciójával az adatbázisban való adatok megváltoztatását. Amennyiben a felhasználó a jelszavát is változtatná és a jelszó követelményeknek szintén megfelel az új jelszó, a regisztrációnál már végzett logika alapján titkosításra kerül sor, majd azt is változtatja az adatbázisban.

Profilkép feltöltése is lehetséges ezen a felületen. A mentéskor először eldönti a setProfile() funkció, hogy a felhasználó tölt-e fel képet, és ha igen, akkor megfelelő-e a formátuma, ezt az isItCorrectImage() funkció dönti el.

```
public function isItCorrectImage(string $imageType) {  
    $allowedTypes = array('image/png', 'image/jpeg', 'image/jpg');  
    if (in_array($imageType, $allowedTypes)) {  
        return true;  
    } else {  
        return false;  
    }  
}
```

Amennyiben megfelelő a formátum, a kép még a szerverre való feltöltés előtt tömörítésre kerül a compress_image() funkció által. A compress_image() funkció elkéri a beküldött képet, a megadott méretre vágja és a megadott minőségre állítja, majd ezt az új képet visszaküldi. Ezek után a visszaküldött, módosított képet szöveges formátumra, később base64-es formátumba kódoljuk, és ezt töltjük fel az adatbázisba a profileDAO megfelelő funkciójával.

```

public function compress_image($source_file, $destination, $quality, $w, $h, $crop=TRUE) {
    //eredeti kép szélessége/magassága/arányának lekérése, és új szélesség/magasság generálása
    list($width, $height, $type) = getimagesize($source_file);
    $r = $width / $height;
    if ($crop) {
        if ($width > $height) {
            $width = ceil($width-($width*abs($r-$w/$h)));
        } else {
            $height = ceil($height-($height*abs($r-$w/$h)));
        }
        $newwidth = $w;
        $newheight = $h;
    } else {
        if ($w/$h > $r) {
            $newwidth = $h*$r;
            $newheight = $h;
        } else {
            $newheight = $w/$r;
            $newwidth = $w;
        }
    }

    //Esetleges elfordulás esetén, álló helyzetbe forgatás
    $info = getimagesize($source_file);
    if ($info['mime'] == 'image/jpeg') {
        $image = imagecreatefromjpeg($source_file);
        $exif = @exif_read_data($source_file);
        if ($exif && isset($exif['Orientation'])) {
            $orientation = $exif['Orientation'];
            if ($orientation != 1) {
                $deg = 0;
                switch ($orientation) {
                    case 3:
                        $deg = 180;
                        break;
                    case 6:
                        $deg = 270;
                        break;
                    case 8:
                        $deg = 90;
                        break;
                }
                if ($deg) {
                    $image = imagerotate($image, $deg, 0);
                }
            }
        }
    }

    elseif($info['mime'] == 'image/png') {
        $image = imagecreatefrompng($source_file);
    }

    //Méretre vágás, minőség csökkentés, új kép készítés
    $dst = imagecreatetruecolor($newwidth, $newheight);
    imagecopyresampled($dst, $image, 0, 0, 0, 0, $newwidth, $newheight, $width, $height);

    imagejpeg($dst, $destination, $quality);

    return $destination;

    //memória felszabadítása
    imagedestroy($image);
    imagedestroy($dst);
    imagedestroy($destination);
}

```

1.8.5.5 Teams felület

A menüszalagon az „Csapatok” kiválasztása esetén, ide irányítja át a felhasználót. Ezen a felületen, azok a csapatok vannak felsorolva, amelyekben a felhasználó csapattag vagy admin. Ezért a teamsController showTeams() funkciója felelős, lekérdezi azoknak a csapatoknak az adatait, amelyeknek a felhasználó a részese, a teamsDAO getTeams() funkciójával. Ezeket az adatokat majd a teams.html.twig template legenerálásakor átküldjük, majd a twig template-n egy „for” ciklussal egyedi linkeket hozunk létre a csapatok egyedi ID-jai alapján.

```
public function loadTeam(int $teamId){
    ... $twig = (new teamsController())->setTwigEnvironment();
    ... $oneTeam = teamsDAO::getOneTeam($teamId);

    ... $userType = (new teamsController())->getUserType($oneTeam);
    ... $userProfilePicture = teamsDAO::getProfilePicture($_SESSION['profPicId']);
    ... $userData = (object) array('username'=>$_SESSION['userName']);

    ... $teamData = (object) array('teamId'=>$teamId);
    ... $membersData = teamsDAO::getTeamMembers($teamId);
    ... $teamName = teamsDAO::getTeamName($teamId);

    ... if($userType=="admin"){
    ...     $_SESSION['groupId']=$teamId;
    ...     echo '<script>alert("admin belepés")</script>';
    ...     echo $twig->render('teams/adminTeam.html.twig', ['userData'=>$userData,
    ...         'teamData'=>$teamData, 'members'=>$membersData,
    ...         'teamName'=>$teamName, 'profPic'=>$userProfilePicture]);
    ... }
    ... elseif($userType=="simpleUser"){
    ...     $_SESSION['groupId']=$teamId;
    ...     echo '<script>alert("simpleUser belepés")</script>';
    ...     echo $twig->render('teams/userTeam.html.twig', ['userData'=>$userData,
    ...         'profPic'=>$userProfilePicture]);
    ... }
    ... elseif($userType=="notUser"){
    ...     header("Refresh:0; url=/showTeams");
    ...     echo '<script>alert("Nem vagy tagja a csapatnak")</script>';
    ... }
}
```

A felület alján, a „CSAPAT LÉTREHOZÁSA” gombra kattintva megjelenik a CreateTeam felület, ahol a csapat profilképe, és neve megadása után a „LÉTREHOZÁS” gombra kattintva a teamsController createTeam() funkciója fut le, mely, a már említett, funkciókkal először leellenőrzi és tömöríti a képet, majd a teamsDAO createTeam() funkciójával felölti az új csapat adatait a szerverre.

Amelyik linkre rákattint a felhasználó, az a csoport fog betölteni. A teamsController loadTeam() funkciója eldönti, hogy a felhasználó az adott csoportban admin-e, tag-e, vagy egyik sem, a teamsDAO getOneTeam() és a teamsController getUserType() funkciók segítségével.

A getOneTeam() funkció lekérdezi, majd visszaküldi egy csapat adminjának ID-ját, csapat tagjainak ID-ját és csapat nevét, melyet a getUserType() funkció felhasznál, és egy „foreach” ciklussal végig nézi a tagokat, és eldönti, hogy a bejelentkezett felhasználó ID-ja megegyezik-e az admin vagy valamelyik tag ID-jával, majd a felhasználó típusának megfelelő szöveges üzenetet küld vissza, melyet a loadTeam() funkció eltárol egy \$userType változóban.

A \$userType változó alapján, dönti el a loadTeam() funkció, hogy melyik template-t kell legenerálnia.

Amennyiben a felhasználó admin akkor az adminTeam.html.twig template tölt be, ahol az admin további felhasználókat tud felvenni a csapatba email-címek segítségével. Ezért a teamsController inviteUserToTeam() funkció felelős, amely az admin által beírt, vesszővel elválasztott email címeket továbbítja a teamsDAO inviteUsers() funkciójának, amely először szétválasztja és egy tömbbe helyezi a beküldött email címeket, majd egy „foreach” ciklussal eldönti, hogy az adott email cím már tag-e. Ha nem, és létezik azzal az email címmel felhasználó, akkor felveszi a csapatba, azaz beírja az éppen megnyitott csapatnak a tagjai közé az adatbázisban.

```
public static function inviteUsers(string $teamUsersEmails, int $teamId){
    .... $dbObj = new Database();
    .... $conn = $dbObj->getConnection();
    .... //csapat tagjainak feltöltése
    .... $userEmails= explode(",", $teamUsersEmails);
    ....
    .... foreach($userEmails as $email){
    ....     //aktuális csapat tag ID email alapján
    ....     $statement=$conn->prepare("SELECT userId FROM users WHERE email=:email;");
    ....     $statement->setFetchMode(\PDO::FETCH_OBJ);
    ....     $statement->execute([
    ....         'email'=>$email,
    ....     ]);
    ....     $userId= $statement->fetch();
    ....
    ....     if($userId!=NULL){
    ....         //már felvett tagok
    ....         $statement=$conn->prepare("SELECT userId FROM groupTags
    ....         WHERE userId=:userId AND groupId=:groupId;");
    ....         $statement->setFetchMode(\PDO::FETCH_OBJ);
    ....         $statement->execute([
    ....             'userId'=>$userId->userId,
    ....             'groupId'=>$teamId,
    ....         ]);
    ....         $isThereAUser= $statement->fetch();
    ....         //nem felvett tagok felvétele
    ....         if($isThereAUser==NULL){
    ....             $statement=$conn->prepare("INSERT INTO groupTags (userId, groupId)
    ....             VALUES (:userId, :groupId);");
    ....             $statement->execute([
    ....                 'userId'=>$userId->userId,
    ....                 'groupId'=>$teamId,
    ....             ]);
    ....         }
    ....     }
    .... }
}
```

Az admin meg tudja jeleníteni az összes csapattag adatait a teamsController loadTeam() funkciója miatt. Még mielőtt betöltődik a megfelelő template, a teamsDAO getTeamMembers() funkció lekérdezi a csapat tagjainak adatait, majd a visszaküldi a loadTeam() funkciónak ami továbbítja majd a template-nek. Így a twig templaten egy „for” ciklussal legeneráljuk a csoport összes tagjának az adatát egy táblázatba.

Az admin az adminTeamStatistics javascript által létrehozott egyedi diagramokat tudja megjeleníteni, a megfelelő gomb megnyomásával. Amelyekhez a megfelelő adatokat a teamsDAO kérdezi le, majd továbbításra kerül, hogy a javascript elérhesse ezeket az adatokat.

Amennyiben a felhasználó csapattag akkor az userTeam.html.twig template tölt be, ahol a felhasználó a megfelelő gomb megnyomásával egyedi diagramokat tud megjeleníteni. Amelyhez az adatokat a már említett módon éri el.

1.8.5.6 Rating felület

A menüszalagon az „értékelés” kiválasztása esetén, ide irányítja át a felhasználót. A felhasználó az értékelés adatainak megadása után a „KÜLDÉS” gombra kattintva a ratingController sendRating() funkciója elküldi az adatokat a ratingDAO setRating() funkciójának, ami feltölti az értékelés adatait az adatbázisba. Amennyiben a felhasználó az értékelése elküldése után újra rálép erre a felületre, a ratingController showRating() funkciója, amely a twig template betöltéséért felelős, leellenőrzi, hogy a felhasználó, küldött-e már értékelést, vagy sem, a ratingDAO decidelfAlreadySent() funkciójával. Ha igen, akkor a showRating() funkció az \$allowRating változót FALSE-ra állítja, majd tovább küldi a twig template-re, és az értékelés le lesz zárva a felhasználónak. Ez azért fontos, hogy egyetlen egy felhasználó ne tudja túlírni az értékeléseket.

1.9 Tesztelés

A tesztelés szintjei a következők:

1. **Unit teszt:** a legkisebb egységek, modulok, objektumosztályok, alrendszerek tesztelése.
2. **Integrációs teszt:** az összeillesztett modulok és interfészek közötti interakciókban keressük a hibákat. Két stratégia terjedt el az integráció során: a fentről lefelé és a lentől felfelé való integrálás. Az egyik esetben a tesztelést a felsőbb szinteken kezdjük, és lefelé haladva keressük a hibákat, míg a másodikonál a tesztelt egységeket illesztjük egymáshoz felfelé haladva a hierarchiában.
3. **Rendszer teszt:** az elkészült rendszer tesztelése annak érdekében, hogy igazoljuk, hogy megfelel a követelményeknek.
4. **Elfogadási teszt:** a megrendelő illetve a felhasználó által megfogalmazott követelmények teljesülését vizsgáljuk. Ez a tesztelési fázis már általában a megrendelő bevonásával történik, 1 dönti el, hogy a rendszer elfogadható vagy sem.
5. **Installációs teszt:** az elfogadás után a rendszert a végső működési környezetbe integráljuk és ott vizsgáljuk a helyes működést.

Ezeket a teszteket végeztük el a teljes programon.

1.9.1 Az asztali alkalmazás tesztjei

Az asztali alkalmazás komponens, integrációs, valamint rendszer testen esett át, melyből némely komponens tesztjét mutatjuk be.

1.9.1.1 Átlagszámítás tesztelése

```
public class Tests {
    readonly ConnectToDB = new ConnectToDB();
    [TestMethod]
    # | 0 references
    public void TestGraphicAverage() {
        .Ddl_dml("TRUNCATE TABLE review;");
        .testInsert("1");
        .testInsert("5");
        double expected = ;
        string szam= .Szamot_ad("SELECT AVG(review.graphics) FROM review");
        double actual = double.Parse(szam);
        Assert.AreEqual(expected, actual);
    }
}
```

Az alábbi osztályban lévő metódus teszteli az átlagszámítást a grafikus felületre:

Ez a metódus a teszt osztálynak tölti fel a táblát megfelelő adatokkal:

```
public void testInsert(string n) {
    Bezaras();
    if(Megnyitas()) {
        try {
            MySqlCommand parancs = new MySqlCommand();
            parancs. = "INSERT INTO review(reviewId,
            graphics, functions, exercises, groups, comments)
            VALUES ('+n+ ", " + n + ", " + n + ", " + n + ", " + n +
            ", '');"
            parancs. = ;
            parancs.ExecuteNonQuery();
            parancs.Dispose();
        } catch(Exception) { }
    }
}
```


1.9.1.2 Szerverkapcsolódás tesztelése

```
public void Megnyitas() {
    try {
        kapcs_mysql = new MySqlConnection(kapcs_string);
        kapcs_mysql.Open();
    } catch (Exception) {
        throw new ServerConnectionException("A szerverhez nem lehet csatlakozni");
    }
}
```

Ezzel a metódussal tudunk kapcsolódni a szerverhez. Abban az esetben ha a szerver kapcsolat nem tud valamilyen okból létrejönni, akkor hibát dob.

A szerver hiba egy egyedileg létrehozott hibatípus a hiba osztályból öröklítve.

```
class ServerConnectionException : Exception {
    1 reference
    public ServerConnectionException(string message) {
    }
}
```

```
public class Test {
    [TestMethod]
    0 references
    public void ConnectionFailed() {
        Assert.ThrowsException<ServerConnectionException>(() => throw
            new ServerConnectionException("A szerverhez nem lehet csatlakozni"));
    }
}
```

Az alábbi teszt, pedig ellenőrzi, hogy a megnyitás során a metódus hibát dob-e.

1.9.1.3 A becslések tesztelése

A teszteléshez szükséges volt egy egy elemmel lefutó függvényt létrehozni a becsléseknél. A függvény

```
public int TestPrediction(MLContext mlContext, ITransformer model, string testComment) {
    PredictionEngine<SentimentData, SentimentPrediction> predictionFunction =
        mlContext.Model.CreatePredictionEngine<SentimentData, SentimentPrediction>(model);
    SentimentData sampleStatement = new SentimentData {
        SentimentText = testComment
    };
    var x = predictionFunction.Predict(sampleStatement);
    return Convert.ToInt32(x.Prediction);
}
```

visszatérési értéke a becsült eredmény, ezért vagy 1, vagy 0 lesz.

```
[TestMethod]
0 references
public void PredictionRight() {
    Predict p = new Predict();
    int actual = p.RunTestPrediction("It was just bad.");
    int expected = 0;
    Assert.AreEqual(expected, actual);
}

[TestMethod]
0 references
public void PredictionRightSecond() {
    Predict p = new Predict();
    int actual = p.RunTestPrediction("I loved it.");
    int expected = 1;
    Assert.AreEqual(expected, actual);
}
```

A tesztelés során mindkét irányba ellenőrizzük a becslést, ezért egy biztosan pozitív és egy biztosan negatív kommentre végezzük el a vizsgálatokat.

1.9.1.4 A regisztráció tesztelése

```
if(key == keyForAuthorization) {
    if(av.Validation() == "felhasznalo hiba") {
        CreateAdmin(username, Sha256(password));
        login.ShowDialog();
    }
} else {
    MessageBox.Show("Nem megfelelő hitelesítési kulcs!!");
    login.ShowDialog();
}
```

A regisztráláshoz szükséges, hogy az adatbázisban ne legyen azonos nevű felhasználó azonos felhasználónévvel.

A tesztnél egy olyan felhasználót hozunk létre, amely még nem létezik az adatbázisban. Ebben az esetben a teszt sikeresen tér vissza.

```
[TestMethod]
public void RegisterNew()
{
    string username = "felhasznalo";
    string password = "admin123!";
    AdminValidation av = new AdminValidation(username,password);
    string expected = "felhasznalo hiba";
    string actual = av.Validation();
    Assert.AreEqual(expected, actual);
}
```

```
[TestMethod]
public void RegisterNew()
{
    string username = "admin";
    string password = "admin123!";
    AdminValidation av = new AdminValidation(username,password);
    string expected = "felhasznalo hiba";
    string actual = av.Validation();
    Assert.AreEqual(expected, actual);
}
```

Az esetet megfordítva, ha olyan felhasználót próbálunk létrehozni, amilyen már létezik az adatbázisban, akkor a teszt hibával tér vissza.

Mivel az adatbázisban a felhasználó jelszava nem ugyanaz, mint a jelentkezés során megadott, ezért jelszó hibával tér vissza a validáció.

```
Message:
Assert.AreEqual failed. Expected:<felhasznalo hiba>. Actual:<jelszo hiba>.
```

1.10 Továbbfejlesztési lehetőségek

1.10.1 Az asztali alkalmazás

Az alkalmazás tovább fejlesztése során lehetőség nyílik arra, hogy a csoportokat is lehessen felügyelni. Ha egy csoporttag, admin nem megfelelő tartalmakat, amelyek bármilyen okból adódóan kifogásolhatóak tesz közzé, a tagok jelenthetik és az adminisztrátorok jogosultak arra, hogy eltávolítsák a fiókját és/vagy megszüntessék a csoportot.

Az adminok ezen felül láthatják a csoportok tevékenységét, így akár jelentés nélkül is kiszűrhetőek a nem megfelelően működő csoportok.

Ezen felül a kihívásokban elküldött videókat AI képfelismerés segítségével szűri, így ha a videó tartalma nem megfelelő az admin törölheti és a felhasználót is törölheti.

Annak érdekében, hogy az admin által törölt felhasználók ne készíthessék el újra a fiókjukat a rendszer automatikusan ellenőrizni fogja, hogy hasonló vagy ugyanazzal a felhasználónévvel vagy e-mail címmel hoztak-e létre fiókot, melyet töröltek. Ha igen az alkalmazás figyelmezteti az admint és megmutatja a hasonló adatokat, amely alapján az admin eldöntheti, hogy törli-e a felhasználót.

Az alkalmazás továbbá képes lesz komplexebb adatelemzésre a bejövő adatok alapján, például, hogy átlagosan mennyi időt töltenek az emberek az alkalmazás használatával.

Az alkalmazás fejlesztése érdekében az adminok képesek lesznek új gyakorlatokat adni az exercise táblához, ezzel bővítve a rendszerben található gyakorlatokat.

Az alkalmazásba a gépi tanulást kiegészítve mesterséges intelligencia kerülne elhelyezésre, amely képes megállapítani, hogy a felhasználók, mivel nincsenek megelégedve, és így eldönteni, hogy mit lehetne fejleszteni, hogyan tovább fejleszteni a megfelelő felhasználó élmény érdekében.

Az alkalmazás valamint elérhető lesz több nyelven is, és a gép nyelvi beállítása alapján automatikusan kiválasztja a nyelvet.

1.10.2 A mobil alkalmazás

A mobil alkalmazás legfőbb továbbfejlesztési iránya, az élő szerverhez való hozzákötés, így a fejlesztők esetleges frissítések keretében új gyakorlatokat adnának hozzá az adatbázishoz, amiket a felhasználó is elérne a saját eszközéről. A szerverrel lehetőség nyílna még a felhasználó adatainak, beállításainak, és legfőképpen az elmentett edzésterveinek feltöltésére, hogy azt az ő adataival, egy másik eszközön is el lehessen érni.

A beállítások menüpontban elérhetővé válik a nyelv kiválasztása, továbbá ide kerül az alkalmazás színösszeállításának megváltoztatásának lehetősége.

Hangeffektusok hozzáadása az alkalmazáshoz, az időzítő váltásához, végéhez, és szüneteléséhez.

A regisztrációs felület, és a statisztikák, felhasználói adatok elérése az androidos alkalmazáson keresztül.

Az alkalmazás portolása IOS és Windows Phone eszközökre is.

Okosórákra fejlesztett verzió.

Feltöltés az Apple Store-ba, a Play Áruházba és az App Gallery-be.

1.10.3 A weboldal

A weboldal továbbfejlesztésének legfontosabb eleme a kihívások funkció, amelyek elsősorban a tanárok, edzők számára lehet segítség a csoportokon belül, mellyel kihívásokat küldhetnek a csoport tagjainak, így kiküszöbölve például a digitális oktatás akadályait. Ezekre a kihívásokra a csapat tagok tudnak majd válaszolni a csapatuk oldalán, a csapattagoknak megjelenített oldalon, hogy elfogadják-e vagy nem. Ha igen, akkor a telefonos alkalmazásban az edzéstervek között megjelenik az a kihívás, amelyet, ha a felhasználó megcsinál egy videót tud küldeni a csoport adminnak bizonyítékként.

Az admin a csapat admin oldalán egy külön gombbal fog tudni megnyitni egy új oldalt, ahol láthatja, hogy ki csinálta már meg az adott kihívást, és egy külön lejátszóban megnézheti a csapattag által küldött videót.

Ezen kívül az admin láthatja ezeknek a kihívásoknak a statisztikáit, mint például, hogy a tagok hány százaléka csinálta már meg.

A csoportok felületen szét lehet majd választani a csapatokat az alapján, hogy ha a felhasználó a csoportban admin, akkor azok a csoportok elkülönítve, az oldal tetején jelenjenek meg, míg, ha tag, akkor az admin csoportjai alatt legyenek megjelenítve.

A felhasználói felület fejlesztés alatt áll, a szerkezete és külseje változhat.

A weboldalra való animációkat lesznek létrehozni, például a bejelentkezéshez, regisztrációhoz, vagy csapatok betöltéséhez.

2 Felhasználói dokumentáció

Készítette: Bakoss Borka, Dutka Lehel Ákos, Ondrik András

2.1 Az asztali alkalmazás

Az asztali alkalmazás háttér logikáját és annak felületét Borka készítette el.

Az alkalmazás azok számára készült, akik felügyelik a mobil alkalmazás és weboldal használatát. Statisztikákat szolgáltat az adatbázis alapján, valamint csoportosítja az értékeléseket.

2.1.1 Rendszerkövetelmények

Operációs rendszer: Windows 10 vagy későbbi 64-bites rendszer

Internet kapcsolat

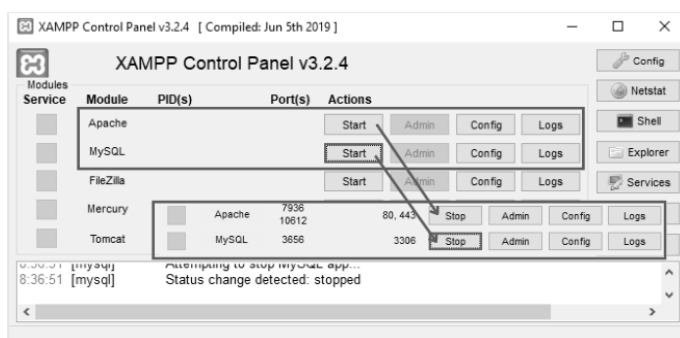
.NET Framework 4.6 vagy későbbi

Minimum 20 MB tárhely

Minimum 10 MB RAM

2.1.2 Telepítési útmutató

1. lépés: A szerver elindítása

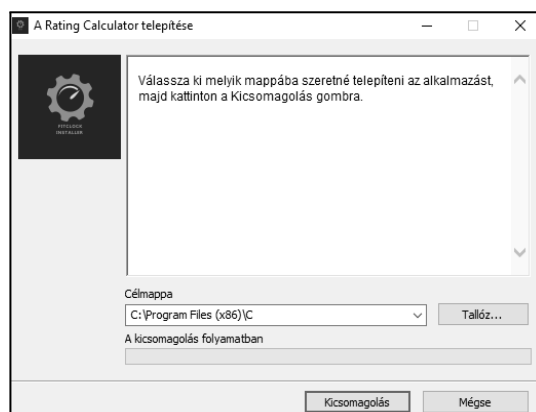


Az alkalmazás használatához szükségünk lesz a szerverre. Az Xampp control paneljén indítsuk el a Mysql és Apache szervert.

2. lépés: Az alkalmazás telepítése

A mellékelt pendrive-on található App.exe fájlt futtassuk. Válasszuk ki, hogy hova szeretnénk telepíteni.

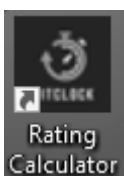
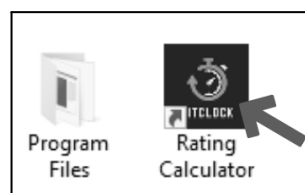
A telepítés után egy Rating Calculator nevű mappát találunk a kiválasztott helyen. Nyissuk meg az Rating Calculator mappát.



3. lépés: Az alkalmazás elindítása

Az alkalmazás és a futtatásához szükséges fájlok a Rating Calculator mappában találhatóak. Az alkalmazás elindításához kattintsunk az App mappában található parancsikonra.

A parancsikon a telepítés során az asztalon is létre lesz hozva.



Megjegyzés: Telepítéskor a Microsoft Defender vagy más vírusirtók veszélyes szoftverként azonosíthatják a programot vagy vírussal fertőzöttként, mivel ismeretlen gyártó számukra a program készítője, ezért nem tartják megbízhatónak.

2.1.3 Felületek

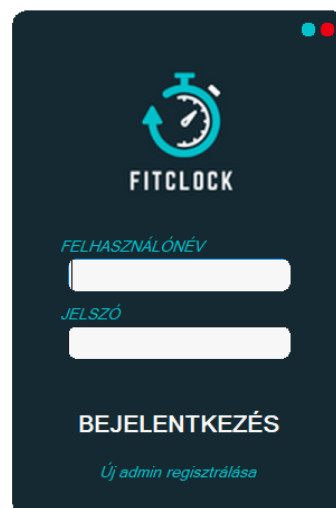
2.1.3.1 Login felület

A bejelentkezéshez két mező kitöltése szükséges. A felső szövegdobozba írjuk be a felhasználó nevünk, a másodikba pedig a jelszavunkat. Karakterek helyett csillagokat jelenít meg, hogy a szöveg titkosítva legyen.

A bejelentkezés gombra ellenőrzi a megadott adatok alapján jogosultak vagyunk-e a belépésre.

Amennyiben még nem csináltunk fiókot, akkor a regisztrálásra kattintva új admint hozhatunk létre.

A jobb felső sarokban lévő gombbal kiléphetünk az alkalmazásból, míg a mellette lévő kék gombbal tálcára tehetjük, minimalizálhatjuk az ablakot.



Sikertelen bejelentkezés!

Helyesen adott meg minden adatot?

VISSZA

Sikeres belépés esetén a Dashboard felületre lépünk tovább, hiba esetén figyelmeztet, hogy nem megfelelőek a belépési adatok, illetve figyelmeztet, hogy regisztráljunk, ha még nincs fiókunk.

2.1.3.2 Dashboard felület

A Dashboard felületen láthatóak az adatbázisból végzett statisztikák.

Az értékeléseknél négy fő szempontot kell értékelni ötös skálán: grafika, gyakorlatok, csoportok funkciói, alkalmazás funkciói. Ezekből kiszámolja az egyes szempontok átlagos értékelését, így látható, hogy az alkalmazás melyik része szorul fejlesztésre.

A felhasználók az értékelésekhez szöveges megjegyzést is fűzhetnek. A megjegyzések gomb az ezzel foglalkozó felületre visz át minket.

A grafikon mutatja a felhasználók számát adott időszakokban. A mérést minden egyes belépésnél frissíti, így mindig a legfrissebb adat is szerepel az adatok közt.

Kiírja az aktuális felhasználók számát szöveggel, valamint kiszámolja a legutóbbi mérés óta való növekedést a felhasználók számában. Amennyiben aznap már volt mérés végezve, akkor a növekedés eredménye nulla lesz.

A jobb felső sarokban lévő nyíllal kiléphetünk a fiókunkból és a login felületre helyez vissza az alkalmazás.

A piros gomb kiléptet fiókunkból és az alkalmazásból, míg a kék gomb tálcára helyezi, minimalizálja az ablakot.



2.1.3.3 Register felület

A sikeres regisztrációhoz három mezőt kell kitöltenünk.

Az első mezőben egy egyedi felhasználónevet kell megadnunk. A második mező tartalmazza az általunk megadott jelszót, mely maximum 14 karakteres lehet.

A harmadik mező tartalmazza hitelesítési kulcsot, mely nélkül nem lehet új fiókot létrehozni. A kulcs nem publikus, ezért csak a már regisztrált adminok ismerik, így elkerülhető, hogy bárki illetéktelenül hozzáférést szerezzen statisztikai adatokhoz.

A regisztrálás gombra kattintva hitelesíti a regisztrációt, és amennyiben sikeres, már azonnal be lehet jelentkezni a fiókba.

REGISZTRÁCIÓ

FELHASZNÁLÓNÉV
admin

JELSZÓ

JELSZÓ MEGERŐSÍTÉSE

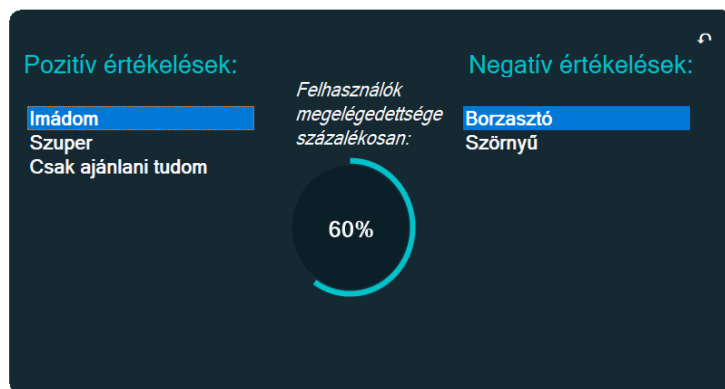
KULCS

REGISZTRÁLOK VISSZA

A regisztrálás gomb csak abban az esetben lesz aktív, ha két mezőben a jelszó megegyezik.

A lent lévő vissza gombbal visszaléphetünk a bejelentkezéshez.

2.1.3.4 Comment felület



A Comment felületen láthatóak az értékelések során kapott megjegyzések.

A baloldali lista tartalmazza a pozitív, míg a jobboldali a negatívnak kategorizált megjegyzéseket. A magyar nyelvre való fordítás miatt a pontossága csökken a beépített Windows.MI funkcióinak, ezért a megjegyzés csak abban az

esetben lesz kategorizálva, ha a rendszer 35%-ig biztos annak a pontosságában. Ellenkező esetben a megjegyzést elveti az alkalmazás és nem használja további statisztikákhoz.

A középső oszlopban a felhasználók megelégedettségét számítja ki a megjegyzések alapján százalékban.

A jobb felső sarokban lévő gombbal visszaléphetünk a Dashboard felületre.

2.2 A mobil alkalmazás

A mobil alkalmazás háttér logikáját és felületét András készítette el.

2.2.1 Rendszerkövetelmények

Operációs rendszer: Android 13 vagy későbbi
Minimum 200 MB tárhely

2.2.2 Telepítési útmutató

1. lépés: Az alkalmazás telepítése

Az alkalmazás telepítéséhez koppintson a képernyőn a FitClock ikonra.

2.2.3 Felületek

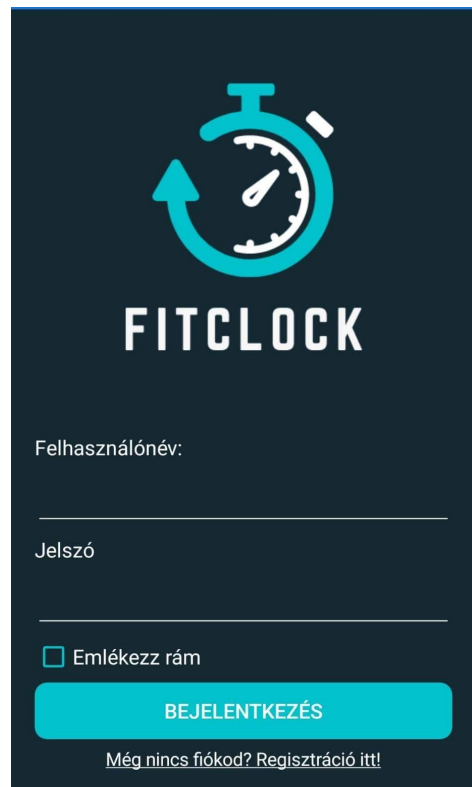
2.2.3.1 Login

Az Androidos alkalmazás megnyitásakor az első képernyő a bejelentkező felület.

Itt az alkalmazás logója alatt, a felhasználónév és jelszó megadására szolgáló szöveg beviteli mezők láthatók. Az emlékezz rám négyzet dönti el, hogy sikeres bejelentkezés után az adatok mentésre kerüljenek e. Ezek az alkalmazás bezárása és újraindítása után ismét megjelennek a beviteli mezőkben.

A bejelentkező gomb ellenőrzi az adatok helyességét, és megfelelő adatok esetén továbbengedi a felhasználót az alkalmazásba.

A gomb alatti link, a regisztrációs weboldalra mutat.

A dark-themed login screen for the FitClock app. At the top center is a teal stopwatch icon with a circular arrow around it. Below the icon, the word "FITCLOCK" is written in white, bold, uppercase letters. Underneath, there are two input fields: the first is labeled "Felhasználónév:" and the second is labeled "Jelszó". Below these fields is a checkbox labeled "Emlékezz rám". At the bottom, there is a large teal button with the text "BEJELENTKEZÉS" in white. Below the button, there is a link that says "Még nincs fiókod? Regisztráció itt!".

Felhasználónév:

Jelszó

☐ Emlékezz rám

BEJELENTKEZÉS

[Még nincs fiókod? Regisztráció itt!](#)

2.2.3.2 Navigáció



Az alkalmazásban a navigáció egy bal oldalról kinyíló menüvel történik. Ezt minden felületről, a bal felső ikonra koppintva hívhatjuk elő.

Ebbe a menübe került még be a kilépés lehetősége is. Erre koppintva egy felugró ablak megerősítést kér, majd kilépteti a felhasználót az alkalmazásból.

2.2.3.3 Edzésterv készítés

Az edzés készítése felületen a felhasználó új edzéstervet hozhat létre amik a lokális SQLite adatbázisban tárolódnak el.

A készítés úgy történik, hogy a felületen látható gyakorlatokra koppintva, egy felugró ablak a feladat idejét, és az utána következő pihenés idejét kéri be másodpercben.

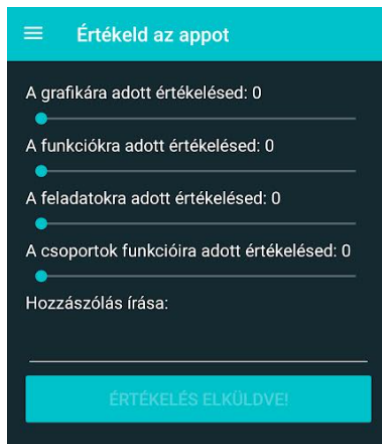
Ezek megadása után a gyakorlat hozzáadódik a készülő edzéstervhez. Ezt folyamatosan frissülve láthatjuk a gyakorlatok listája alatt, vesszőkkel elválasztva.

A legutoljára hozzáadott gyakorlatot egy erre kijelölt gombbal eltávolíthatjuk az edzéstervből.

Az kész gombra koppintva egy felugró ablak bekéri az elkészült edzésterv nevét, amit azzal a névvel elment. Végül átnavigál a edzésterv listája felületre.

Edzés készítése	
Burpee	Nehézség: 5
Hegymászó	Nehézség: 5
Lebegő lábak	Nehézség: 5
Plank	Nehézség: 5
Hasprés	Nehézség: 5
Fekvőtámasz	Nehézség: 5
Tolódzkodás	Nehézség: 5
Lábemelés	Nehézség: 5
Magas térdemelés	Nehézség: 5
KÉSZ	

2.2.3.4 Értékelés felület



Ezen a felületen a felhasználó visszajelzést küldhet az alkalmazásról.

Ezt 4 szempont alapján, 1-től 10-ig értékelheti csúszkák használatával.

Lehetőség van egy többsoros hozzászólás beírására is.

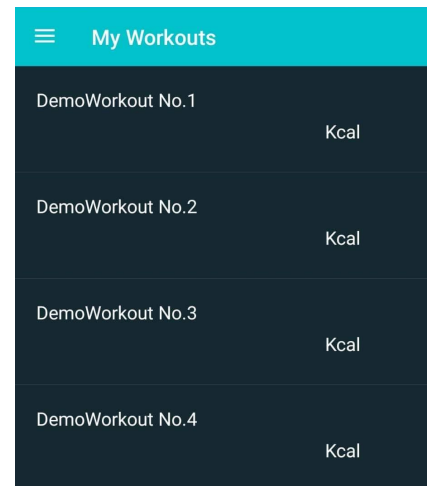
Végül a beküldés gombbal az adatok felkerülnek egy adatbázisba. A felület pedig lezárul, így egy felhasználó, csak egy értékelést küldhet be.

2.2.3.5 Edzésterv lista

A belépés utáni első felület az edzésterv lista. Itt a felhasználó által létrehozott, SQLite adatbázisban tárolt edzéstervek jelennek meg.

Az edzéstervekre rányomva az alkalmazás átnavigál az edzés felületre.

Hosszan nyomva a kiválasztott edzésre, a felület tetején megjelenő gombbal kitörölhetjük azt az adatbázisból.



My Workouts	
DemoWorkout No.1	Kcal
DemoWorkout No.2	Kcal
DemoWorkout No.3	Kcal
DemoWorkout No.4	Kcal

2.3 A weboldal

A weboldal háttér logikáját és annak felületét Lehel készítette el.

A weboldal azok felhasználók számára készült, akik szeretnének csapatokban dolgozni és bővebb statisztikákat látni. Lehetőséget kínál csapatok létrehozására, statisztikai diagrammok követésére és az alkalmazások értékelésre is.

2.3.1 Rendszerkövetelmények

Operációs rendszer: Windows 10 és későbbi Operációs rendszerek

Böngésző: általános, naprakész böngészők

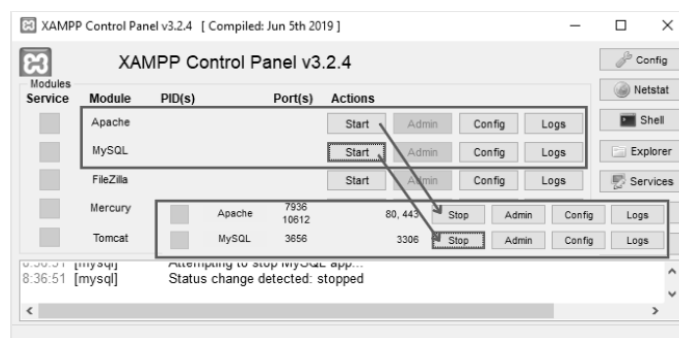
Ajánlott böngésző: Opera vagy Chrome

Ajánlott képernyő méret: 15"

Ajánlott képernyő felbontás: 1920x1080

2.3.2 Telepítési útmutató

1. lépés: A szerver elindítása



A weboldal használatához szükségünk lesz a szerverre. Az Xampp control paneljén indítsuk el a Mysql és Apache szert.

2. lépés: A konzolon való webszerver elindítása lokálisan.

Megkeressük a konzolon azt a mappát, amiben a weboldal elhelyezkedik.

Az elérési út ebben az esetben: A:\XAMPP\htdocs\fitclock.

Itt kiadjuk a „composer update” parancsot, hogy a legfrissebb technológiákkal dolgozzunk, majd a „php -S localhost:8000” parancsot, hogy elinduljon a lokális webszerver a localhost:8000 porton.

```
A:\>cd XAMPP
A:\XAMPP>cd htdocs
A:\XAMPP\htdocs>cd fitclock
A:\XAMPP\htdocs\fitclock>composer update
Loading composer repositories with package information
Info from https://repo.packagist.org: #StandWithUkraine
Updating dependencies
Nothing to modify in lock file
Installing dependencies from lock file (including require-dev)
Nothing to install, update or remove
Generating autoload files
3 packages you are using are looking for funding.
Use the `composer fund` command to find out more!
No security vulnerability advisories found
A:\XAMPP\htdocs\fitclock>php -S localhost:8000
[Wed Apr 12 12:38:59 2023] PHP 8.2.0 Development Server
(http://localhost:8000) started
```

3. lépés: A böngészőben megnyitni a weblapot, a localhost:8000 porton.

A böngésző címsorába beírjuk, hogy „localhost:8000”. Ezt betöltve elérjük a weboldalunkat.



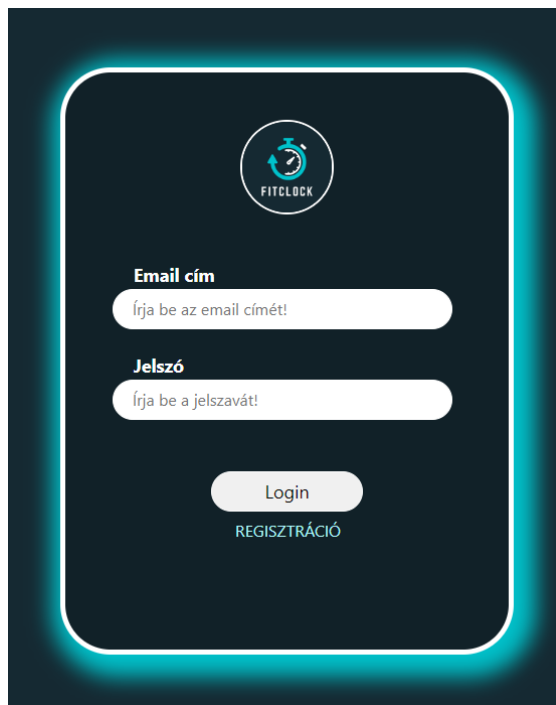
2.3.3 Felületek

2.3.3.1 Login felület

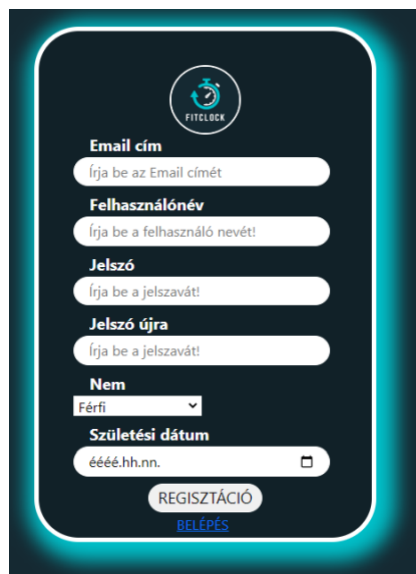
A bejelentkezéshez az első mezőbe ber kell írunk a felhasználónevet és másodikba a jelszót.

Amennyiben megfelelő adatokat adtunk meg, akkor továbbít minket a weboldal többi részére. Ellenkező esetben figyelmeztet minket a hibás adatra.

A bejelentkezés gomb alatt található regisztráció gomb segítségével új fiókot hozhatunk létre.



2.3.3.2 Regisztráció felület



Az új fiók létrehozásához szükséges megadnunk egy e-mail címet, egy egyedi felhasználónevet, egy erős jelszót. Ezeket az adatokat később is megváltoztathatjuk.

Egy e-mail címmel egyszerre csak egy fiókot hozhatunk létre. Ha az adatbázisban már létezik ilyen e-mail címmel fiók, akkor figyelmezteti a felhasználót.

A jelszót ellenőrzi, hogy legalább 6 karakter hosszú, kis és nagybetűt, valamint számot is tartalmazzon.

A nemünket kiválaszthatjuk egy drop-down listából.

A legutolsó adat a születési adat, melyet akár gépelve, akár naptárból kiválasztva megadhatjuk.

2.3.3.3 Beállítások felület

A regisztráció során megadott adatokat módosíthatjuk, frissíthetjük.

Itt adhatjuk meg a statisztikák pontosságához szükséges adatokat, mint a testsúly, a magasság, valamint új profilképet adhatunk meg.

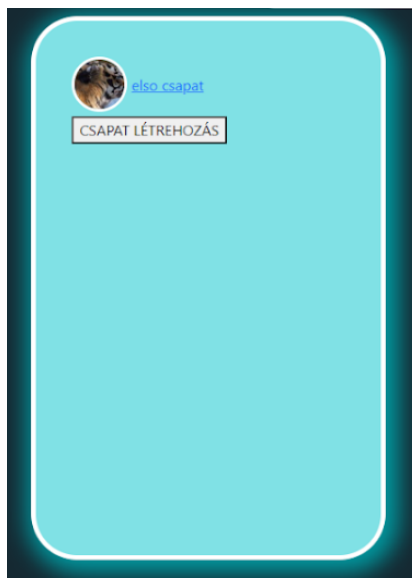
Fontos, hogy a módosítások csak akkor történnek meg, ha a mentés gombra kattintunk, különben az oldalrészt elhagyva elvesznek.

A jelszót szintén ellenőrzi, hogy legalább 6 karakter hosszú, kis és nagybetűt, valamint számot is tartalmazzon.



A screenshot of a user profile settings form. The form is titled "profilkép feltöltése" and includes a "Fájl kiválasztása" button and a status "Nincs fájl kiválasztva". Below this is the "Felhasználónév" field with the value "Newuser1". The "Súly(kg)" field is empty. The "Magasság(cm)" field is empty. The "Új jelszó" field is empty. The "Új jelszó újra" field is empty. The "Születési dátum" field shows "1989.01.01." with a calendar icon. At the bottom is a "Mentés" button.

2.3.3.4 Csoportok felület



A screenshot of a team management interface. It shows a team profile picture placeholder with a small circular icon and the text "első csapat". Below this is a button labeled "CSAPAT LÉTREHOZÁS".

Ezen a felületen kezelhetjük az aktuális csoportjainkat, valamint új csoportot hozhatunk létre, vagy csatlakozhatunk egyhez.

A meglévő csoport nevére kattintva léphetünk be az adott csoportba.

A csoportban, ha adminok vagyunk láthatjuk a csoporttagokra vonatkozó statisztikákat, illetve kihívásokat küldhetünk az egész csoportnak.

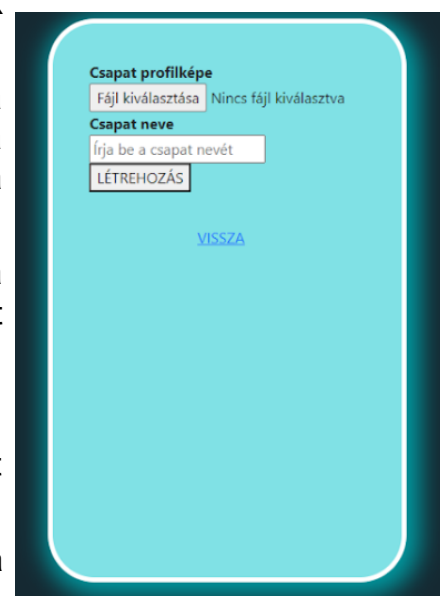
A Csoport létrehozása gombra kattintva hozhatjuk létre a csoportot.

A Fájl kiválasztása gombra kattintva feltölthetünk egy képfájlt a csoport profilképéként.

A szövegdobozban megadhatjuk a csoport nevét.

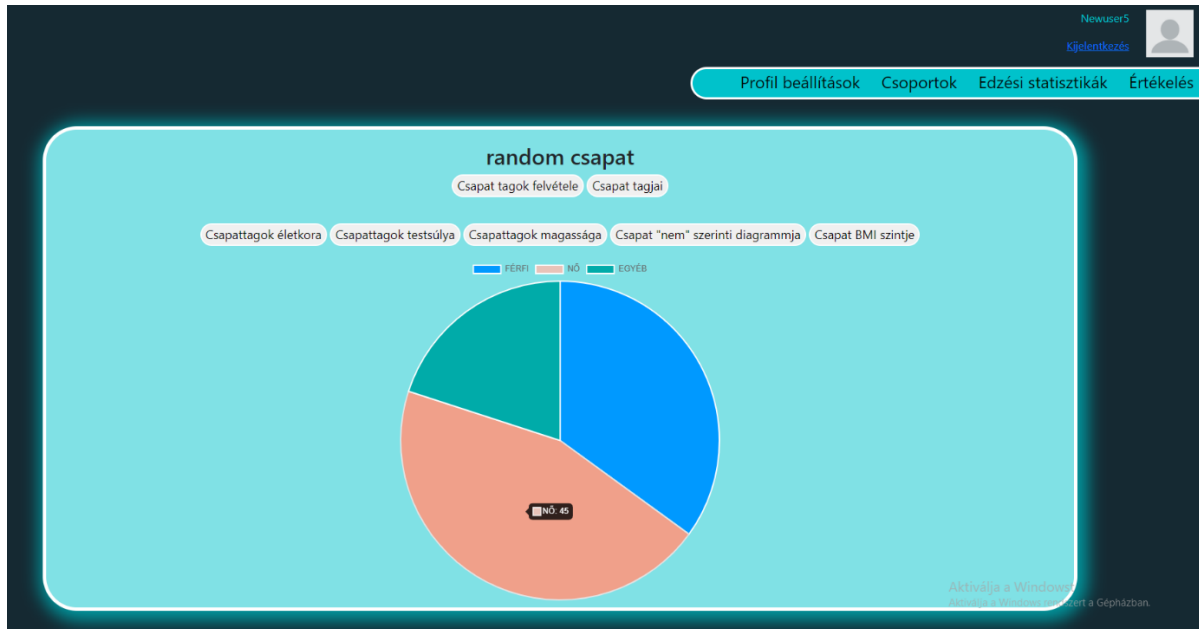
A Létrehozás gombra kattintva létrejön a csoport melynek a felhasználó lesz az adminja.

Ha mégse szeretnénk új csoportot létrehozni a vissza gombbal a csoportjainkhoz lépünk vissza.



A screenshot of a team creation form. It is titled "Csapat profilképe" and includes a "Fájl kiválasztása" button and a status "Nincs fájl kiválasztva". Below this is the "Csapat neve" field with the placeholder text "Írja be a csapat nevét". At the bottom is a button labeled "LÉTREHOZÁS".

A csoportban, ha adminok vagyunk láthatjuk a csoporttagokra vonatkozó statisztikákat. Gombok segítségével nyithatunk meg popup ablakokat ahol felvehetünk további tagokat, vagy láthatjuk az összes csapattag adatait és ha az admin úgy dönt, ki is rúghatja őket.

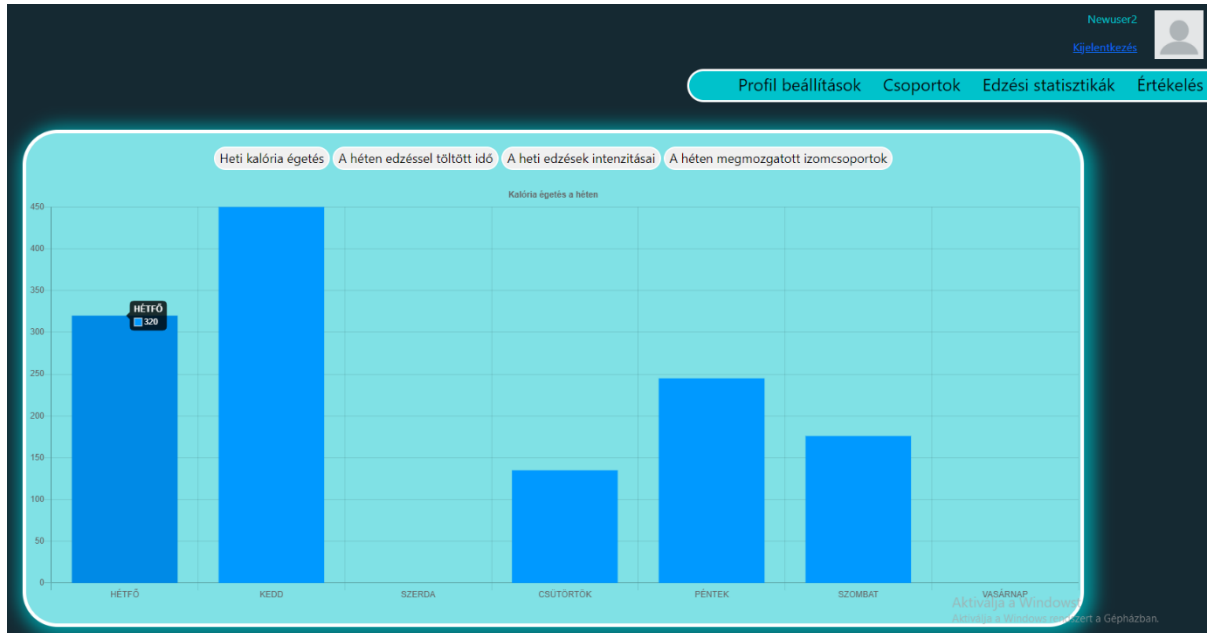


A csapatban, ha tagok vagyunk láthatjuk a csapat és saját statisztikánkat.



2.3.3.5 Statisztikák felület

Láthatjuk a héten történt edzéseinkre vonatkozó statisztikákat. A különböző gyakorlatokkal láthatjuk, milyen izomcsoportokra edzettünk. Láthatjuk a héten elégetett kalóriák számát, a heti edzések intenzitását, valamint, hogy a héten melyik nap mennyi időt töltöttünk edzéssel.



2.3.3.6 Értékelés felület

Grafika, az oldalunk kinézete:
1-Rossznak tartom
Funkciók, amikre az oldalunk képes
2-Nem vagyok megelégedve
Edzések, amiket az alkalmazásunk kínál
3-Meg vagyok elégedve
Csoportok, a csoportjaink működése
4-Jónak tartom
További megjegyzés:
Ide írja a megjegyzését...
KÜLDÉS

Ha szeretnénk, értékelést küldhetünk a rendszer fejlesztése érdekében.

Négy szempont alapján egy 5-ös skálán értékelhetünk.

Amennyiben szeretnénk szövegesen is értékelhetjük, valamint kritikát is megfogalmazhatunk a programokkal kapcsolatban.

Ha egyszer kitöltöttük, akkor már nem tudjuk még egyszer kitölteni.

3 Irodalomjegyzék

Reiter István:

C# programozás lépésről lépésre (2012)

Horváth László:

Szoftvertesztelés a gyakorlatban (2014)

Esposito Dino:

Programming ML.Net (2022)

Christian Nagel:

Professional C# and .NET - 2021 Edition (2021)

Dan Hemes:

Xamarin Mobile Application Development (2015)

Jon Dockett:

PHP & MySQL - Server-side Web Development (2018)

Jon Dockett:

Web Design with HTML, CSS, JavaScript and jQuery Set (2014)

Robert C. Martin:

Tiszta kód (2008)

Douglas Squirrel és Jeffrey Fredrick:

Agile Conversations (2020)

Mark Massé:

REST API Design Rulebook (2011)

Rudolf Pecinovský:

OOP - Learn Object Oriented Thinking and Programming (2013)

Használt oldalak:

<https://learn.microsoft.com/>

<https://www.optimizely.com/>

<https://dotnet.microsoft.com/>

<https://httpd.apache.org/>

<https://www.oracle.com/mysql/>

<https://getcomposer.org/>

<https://www.php.net/>

<https://www.tutorialspoint.com/javascript/>

<https://twig.symfony.com/>

<https://learn.microsoft.com/hu-hu/xamarin/>

<https://stackoverflow.com/>

<https://www.w3schools.com/>

<https://visualstudio.microsoft.com/>

<https://code.visualstudio.com/>

<https://promanconsulting.hu/>

4 Mellékletek

Mappák (github ágak):

- dokumentacio
Tartalmazza a dokumentáció digitális verzióit.
- app
Az asztali alkalmazást telepítő fájlt tartalmazza
- mobil
A mobil alkalmazást telepítő fájlt tartalmazza
- web
A weboldal fájljait, szkriptjeit, kódjait tartalmazza
- server
Az Xampp telepítő fájlt tartalmazza és az sql fájlokat.

github link:

<https://github.com/bssborka/Szakdolgozat>