Szegedi Szakképzési Centrum Vasvári Pál Gazdasági és Informatikai Szakgimnáziuma

Az 54 213 05 számú Szoftverfejlesztő szakképesítés záródolgozata

Készítette:

Rózsa István

Szeged

2020

Tartalom

[**1.** **Bevezetés** 3](#_Toc36739524)

[**2.** **Fejlesztői Dokumentáció** 4](#_Toc36739525)

[**2.1 Adatbázis** 4](#_Toc36739526)

[**2.1.1 Adatbázis ábra** 4](#_Toc36739527)

[**2.1.2 Egyedek és tulajdonságaik** 4](#_Toc36739528)

[**2.2 Asztali alkalmazás** 6](#_Toc36739529)

[**2.2.1 Felhasználok** 7](#_Toc36739530)

[**2.2.2. Az alkalmazás** 8](#_Toc36739531)

[**2.2.3 Tesztelés** 10](#_Toc36739532)

[**2.3 Web alkalmazás** 11](#_Toc36739533)

[**2.3.1 Index** 11](#_Toc36739534)

[**2.3.2 Csatlakozás az adatbázishoz** 12](#_Toc36739535)

[**2.3.3 Bejelentkezés és regisztráció** 12](#_Toc36739536)

[**2.3.4 Profil** 13](#_Toc36739537)

[**2.3.5 Reszponzivitás** 16](#_Toc36739538)

[**2.4 Fejlesztési tervek** 16](#_Toc36739539)

[**3. Felhasználó Dokumentáció** 16](#_Toc36739540)

[**3.1 Asztali alkalmazás** 16](#_Toc36739541)

[**3.1.1 Program indítása** 16](#_Toc36739542)

[**3.1.2 Belépés a programba** 17](#_Toc36739543)

[**3.1.3 Üdvözlő ablak** 18](#_Toc36739544)

[**3.1.4 Ételek menüpont** 18](#_Toc36739545)

[**3.1.5 Étrend menüpont** 19](#_Toc36739546)

[**3.2 Web alkalmazás** 20](#_Toc36739547)

[**3.2.1 Oldalak** 20](#_Toc36739548)

[**3.2.2 Funkciók** 20](#_Toc36739549)

[**3.2.2. Menüpontok** 23](#_Toc36739550)

# **Bevezetés**

Egyre nagyobb szerepet játszik a sport, az egészséges életmód az emberek életében. Ezért a záródolgozatomba egy olyan alkalmazást hoztam létre, amely a testépítő életmódot támogatja. A mai világban már majdnem minden dolgát az ember online, interneten keresztül intézi. Azonban ez nem igaz az élet minden területére.

Sportolói pályafutásom során számtalan problémát fedeztem fel. Sokan úgy tartják, hogy a testépítés az nem egy sport. Számtalanszor látok most kezdő hétköznapi embereket az edzőteremben, akik füzettel és tollal a kezükben vágnak bele ebbe a sportba, hogy majd edzéstervet és étrendet írnak maguknak. De sokan nem tudják mikor, mennyit és mit egyenek vagy, hogy milyen gyakorlatokat csináljanak a fejlődés érdekében. Ezek a papírok időközben akár el is tűnhetnek, elkeveredhetnek vagy elmaszatolódhat a rajtuk lévő tartalom ebből adódóan már nem is biztos, hogy olvasható így felesleges időtöltés volt ezeknek a megírása. Nem ez a legmegbízhatóbb formája egy nyilvántartásnak.

Úgy gondolom, hogy a mindennapokat megkönnyítené egy olyan alkalmazás, amely útmutatást, irányt adna akár kezdő, újrakezdő vagy már régóta edző ember számára. Mivel már én is közel 5 éve sportolok egy-két kisebb kihagyással, így tudom, hogy egy ilyen szoftverrel könnyebbek lennének a napok. Egy olyan alkalmazással, amely megvalósítja azokat a funkciókat, aminek a segítségével minden állandósított adat visszanézhető legyen, és nagyobb biztonságot nyújtson, mint a papír és társai. Pontosan ezért választottam ezt a témát, hogy minden téren modern legyen ez a sport.

Ezért úgy döntöttem, hogy megvalósítom ezt az alkalmazást, amely képes arra, hogy felvegye a versenyt a korszerűtlen eszközökkel. A **BuildYourBody** egy olyan korszerű és biztonságos alkalmazás, amellyel minden ember napja egyszerűbbé válik. A rendszeren belül megtalálható számos funkcionális követelmény, mint például minden regisztrált felhasználó BMI-t[[1]](#footnote-1)1 tud majd számolni, meg tudja saját magának határozni a testtömege alapján, hogy egy nap hány gramm fehérjére, szénhidrátra, zsírra van szüksége a fejlődés érdekében. Képes lesz naplózni az étkezéseit, edzéseit és a használt táplálékkiegészítőket. A szoftver megvalósításával minden kliens egy felhasználóbarát, letisztult rendszerben tud majd dolgozni.

# **Fejlesztői Dokumentáció**

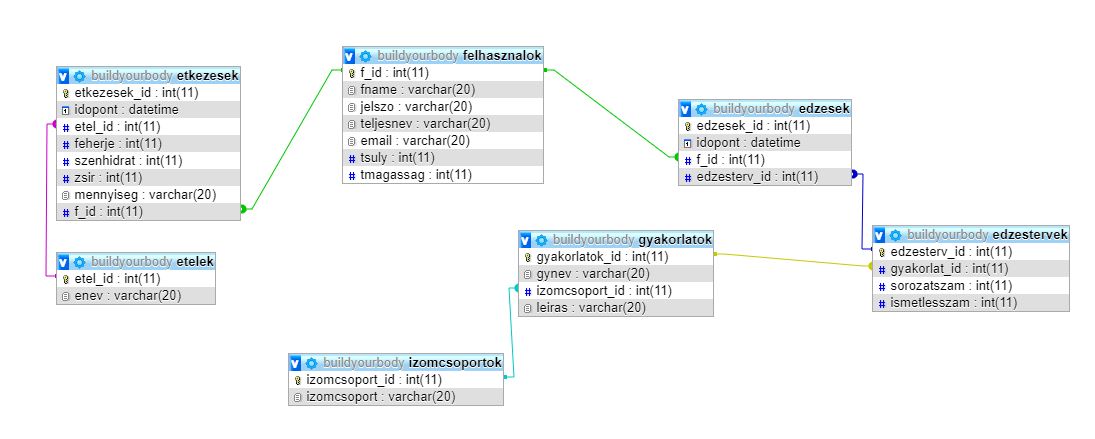
## **2.1 Adatbázis**

A rendszer működéséhez szükség van egy olyan adatbázisra, amely tárolja mindazon adatokat, amelyek előfordulnak egy edzés és életmód során. Az adatbázis a szoftver mindkét részét az asztali és a webalkalmazást is egyaránt kiszolgálja.

Az adatbázis célja, hogy személyre szabottan minden felhasználónak biztosítson bejelentkezést asztali és webes felületen is. Valamint biztosítson webes felületen regisztrációt, edzések összeállítását, valamint gyakorlatok megtekintését. Asztali felületen garantálja az étkezések összeállítását és az ételek megtekintését.

### **2.1.1 Adatbázis ábra**

Ez az ábra szemlélteti az egyedek közötti kapcsolatot, a táblák tartalmát és az adattípusokat.



1. ábra Bachmann-ábra

### **2.1.2 Egyedek és tulajdonságaik**

A **felhasználók** táblába kerül eltárolásra a bejelentkezéshez szükséges felhasználónév és jelszó, a felhasználó teljes neve, e-mail címe valamint testsúlya és testmagassága. Az edzések táblába található személyre szabottan az edzések időpontja és maga az edzésterv. Az **edzéstervek** táblába megtalálható, hogy egy edzéstervben egy gyakorlatból hány sorozatot és hány ismétlést végzünk. A **gyakorlatok** táblában maga a gyakorlat neve, milyen izomcsoporthoz tartozik és egy rövid leírás található. Az **izomcsoportok** táblában fellelhető minden izomcsoport. Az **étkezések** táblában látható egy étkezés időpontja, maga az étel és a mennyisége, ami elfogyasztásra kerül, és hogy mennyi fehérjét, szénhidrátot és zsírt vittünk be a szervezetbe az étkezéssel. Az **étel** táblában található minden olyan étel, amelyek az étkezések során fogyasztásra kerülnek.

A **felhasználók** tábla tulajdonságai:

* f\_id: Egész szám, elsődleges kulcs, automatikusan növekvő érték, amely a felhasználók megkülönböztetésére szolgál.
* fname: Szöveg, a bejelentkezéshez szükséges felhasználónév, egyedi.
* jelszo: Szöveg, a bejelentkezéshez szükséges jelszó.
* teljesnev: Szöveg, a felhasználó neve.
* email: Szöveg, a felhasználó e-mail címe.
* tsuly: Egész szám, a felhasználó testsúlya.
* tmagassag: Egész szám, a felhasználó testmagassága.

Az **edzések** tábla tulajdonságai:

* edzesek\_id: Egész szám, elsődleges kulcs, automatikusan növekvő érték, az edzés azonosítója.
* idopont: Date típusú, az edzés időpontját jelöli, automatikusan növekvő érték (curdate).
* f\_id: Egész szám, idegen kulcs, ebből az értékből tudjuk meg, hogy melyik felhasználóhoz tartozik az adott edzés.
* edzesterv\_id: Egész szám, idegen kulcs, megmutatja, hogy az adott edzések milyen gyakorlatok kerültek elvégzésre.

Az **edzéstervek** tábla tulajdonságai:

* edzesterv\_id: Egész szám, elsődleges kulcs, automatikusan növekvő érték, az edzésterv azonosítója.
* gyakorlat\_id: Egész szám, idegen kulcs, a gyakorlat azonosítója.
* sorozatszam: Egész szám, megmutatja, hogy az adott gyakorlatból hány sorozat kerül elvégzésre.
* ismetlesszam: Egész szám, megmutatja, hogy az adott gyakorlatból hány ismétlés kerül elvégzésre egy sorozaton belül.

A **gyakorlatok** tábla tulajdonságai:

* gyakorlatok\_id: Egész szám, elsődleges kulcs, automatikusan növekvő érték, a gyakorlat azonosítója.
* gynev: Szöveges típusú, a gyakorlat neve.
* izomcsoport\_id: Egész szám, idegen kulcs, az izomcsoport azonosítója, amelyhez a gyakorlat tartozik.
* leiras: Szöveges típusú, információ a gyakorlatról.

Az **izomcsoportok** tábla tulajdonságai:

* izomcsoport\_id: Egész szám, elsődleges kulcs, az izomcsoport azonosítója.
* izomcsoport: Szöveges típusú, az izomcsoport megnevezése.

Az **étkezések** tábla tulajdonságai:

* etkezesek\_id: Egész szám, elsődleges kulcs, automatikusan növekvő érték, az étkezés azonosítója.
* idopont: Datetime típusú érték, megmutatja, hogy mikor volt az adott étkezés.
* etel\_id: Egész szám, idegen kulcs, az étel azonosítója, amely fogyasztva volt az étkezésen belül.
* f\_id: Egész szám, idegen kulcs, a felhasználó azonosítója, akihez az étkezés tartozik.

Az **ételek** tábla tulajdonságai:

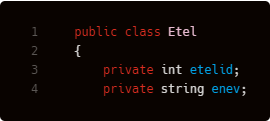
* etel\_id: Egész szám, elsődleges kulcs, automatikusan növekvő érték, az étel azonosítója.
* enev: Szöveges típusú, az étel neve.
* szenhidrat: Egész szám, az étel szénhidrát tartalma.
* feherje: Egész szám, az étel fehérje tartalma.
* zsir: Egész szám, az étel zsír tartalma.
* kaloria: Egész szám, az étel energia tartalma.
* mennyiség: Szöveges típusú, az adott mennyiség az ételből, amelyben megtalálható a fehérje, szénhidrát, zsír és kalória mértéke.

## **2.2 Asztali alkalmazás**

Az asztali alkalmazás elkészítéséhez egy olyan programozási nyelvet választottam, amiben már jártas vagyok. Ezért a választás a C#-ra esett. Az iskolai tanulmányaim során is a legtöbb projektet e nyelv segítségével oldottuk meg, így rögtön ezt választottam. Nagyon fontos, hogy ezt a programozási nyelvet egy platformon lehet használni, nem platformfüggetlen mint például a Java. A fejlesztői környezet kiválasztása sem okozott gondot, a legoptimálisabb a Microsoft Visual Studio. Mivel egyszerű kezelni, ingyenes, könnyen személyre lehet szabni a felhasználói felületet és rengeteg ingyenes keretrendszer letölthető hozzá. Ennek a programnak a 2019-es verzióját használtam. Viszont ami a legfontosabb, hogy a Visual Studio képes kapcsolatot létesíteni olyan verziókövető rendszerrel, mint például a GitHub, és felhőben is képes eltárolni a program aktuális állapotát. Egy ilyen nagy és hosszadalmas munkánál elengedhetetlen a verziókövető rendszerek használata.

A program fejlesztése során igyekeztem olyan környezetet kialakítani, amely felhasználóbarát és könnyű kezelhetőséget biztosít. A fejlesztés során pedig megmaradni az objektum orientált programozás mellett, így hát nem újra és újra írtam a kódokat, hanem mindenre metódust készítettem, amiket többször is meg lehet hívni. Így felesleges kódsorokat spóroltam meg.

Az objektum orientált programozás olyan módszereket nyújt nekünk fejlesztőknek, amelyek lehetővé teszik a programok bonyolultságának csökkentését, a megbízhatóság és a hatékonyság növelését. Az OOP objektumokból építi fel a programot. Az objektum adatokat tárol, és kérésre tevékenységet végez.

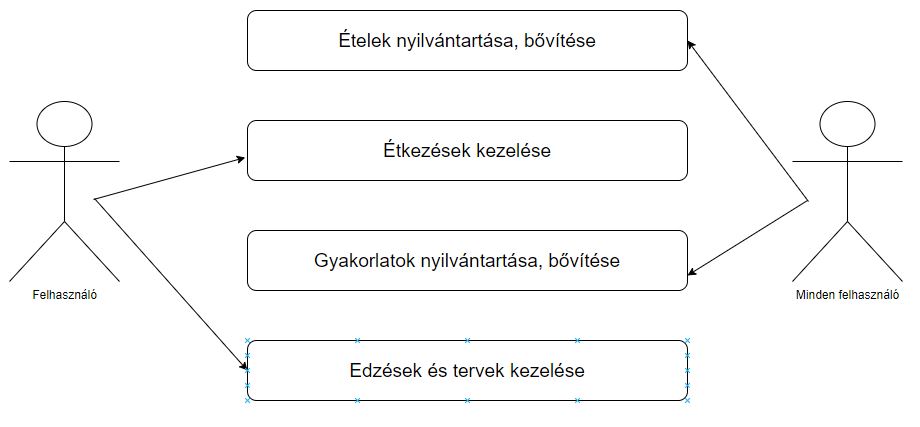


2. ábra Objektum a hozzátartozó attribútumokkal

A felhasználói felület minőségét befolyásolja a felhasznált színek száma és fajtája valamint, az olvashatóság, így igyekeztem olyan ablakokat létrehozni ahová színes elemeket tettem fel, így sokkal áttekinthetőbb és vonzóbb lett. Ugyanakkor nem használtam túl világos színeket, mivel ezek a színek egy idő után nem tesznek jót az ember szemének.

### **2.2.1 Felhasználok**

A felhasználók feladatainak szemléltetésére az USE-CASE diagramot használjuk. A használat eset diagram egy olyan diagram, amely bemutatja a fejlesztő által elkészített program felhasználóinak a feladatköreit. A mi esetünkben ez a következőképpen néz ki:



3. ábra USE-CASE diagram

Ahogy az ábra is szemlélteti, vannak olyan feladatkörök, amit minden felhasználó egyaránt el tud végezni. Azonban van olyan feladat, amit csak egy felhasználó tud elvégezni. Például Józsi szeretné látni egy másik felhasználó étkezéseit. Azonban erre nincs lehetőség ugyanis az étkezés funkció személyre szabottan van elkészítve.

### **2.2.2. Az alkalmazás**

Mivel egy olyan rendszerről beszélünk, amely személyre szabottan működik, így elengedhetetlen az azonosítás, ahogy a legtöbb program esetében is szükséges, hogy a felhasználó jogosult-e a program használatára. Erre egy olyan ablakot készítettem, ahol a felhasználónév és jelszó megadása után kap egy visszaigazolást a rendszertől, hogy van-e jogosultsága bejelentkezni vagy nincs.

Amint a felhasználó rákattint a bejelentkezés gombra, azonnal lefut egy kérés az adatbázis felé, amely a Login osztályon belül található, ez a kérés ellenőrzi, hogy létezik-e a megadott felhasználónév és jelszó. Ha nem létezik a felhasználónév vagy valamelyik adatot rosszul adott meg a kliens, akkor egy hibaüzenet jelenik meg és a rendszer nem lép. Azonban ha a kliens mindent helyesen adott meg akkor az ,,Sikeres Bejelentkezés” üzenet után a program átirányítja a fő menüre.

Amennyiben sikeres volt a bejelentkezés azt követően a felhasználó előtt megjelenik egy úgynevezett üdvözlő form. Itt lehetőség nyílik a testtömegindex kiszámítására, valamint a menüsávból egy kattintással elérhetővé válnak az ételek vagy az étkezéseink.

Ha szeretnénk megtekinteni az ételeket, akkor válasszuk az ételek menüpontot, majd megjelenik egy új ablak, ami után elénk tárulnak egy ComboBoxban az ételek. Az étel kiválasztása után egy DataGridView-ban láthatóvá válnak az étel adatai.

Az étel hozzáadása gombra kattintva új ételt lehet hozzáadni az adatbázishoz. Ha minden mező helyesen került kitöltésre, akkor megtörténik a sikeres hozzáadás.

if (!ujEtel.validate())

{

return;

}

else

{

//Beszúrás listába

r.addEtelToList(ujEtel);

//Beszúrás adatbázisba

r.insertEtelToDatabase(ujEtel);

//Frissít DataGridView

beallitDataGridView();

FormSucces fs = new FormSucces("Étel mentése sikeres!");

DialogResult result = fs.ShowDialog();

if (result == DialogResult.OK)

{

panelEtel.Visible = false;

fs.Hide();

feltoltComboboxEtelek();

}

}

Amennyiben törölni szeretnénk az ételek közül, úgy az étel törlés gomb kiválasztása után ellenőrzésre kerül, hogy a DataGridView-ban van-e kiválasztva sor. Ha van, akkor a program megkérdezi a klienst, hogy biztosan szeretne-e törölni.

FormMessage fm = new FormMessage("Biztos törölni szeretnél?");

DialogResult result = fm.ShowDialog();

if (result == DialogResult.OK)

{

fm.Hide();

try

{

//Törlés adatbázisból

r.deleteEtelFromDataBase(Convert.ToInt32(dataGridViewEtelek.SelectedRows[0].Cells[0].Value));

//Törlés listából

r.deleteEtelFromList(Convert.ToInt32(dataGridViewEtelek.SelectedRows[0].Cells[0].Value));

}

catch(RepositoryException ex)

{

Debug.WriteLine(ex.Message);

FormError fe = new FormError("Sikertelen törlés, az étel tagja egy étkezésnek!");

DialogResult dr = fe.ShowDialog();

if (dr == DialogResult.OK){

fe.Hide();

}

}

//DataGridView frissítés

feltoltComboboxEtelek();

beallitDataGridView();

Ha a törölni kívánt ételt egy étkezés tartalmazza, ez jelzésre kerül a felhasználó felé és ebben az esetben az étel nem törölhető.

Étkezéshez való hozzáadás a megfelelő menü kiválasztásával történik. Ha megtörtént erre a gombra a kattintás, megjelenik egy új panel, amin ki kell, válasszuk az étkezés időpontját. Amint sikerült kiválasztani az időpontot úgy az étel adatai kiolvasásra kerülnek az adattáblából.

Az étkezések a kezdő formon az étrend gombra kattintás után válnak elérhetővé. Egy ComboBoxból kell kiválasztanunk az ott megjelenített dátumot, amelyek az ablak betöltődésekor a konstruktorban kerülnek feltöltésre. A dátum kiválasztása után válnak láthatóvá az étkezéseink. A bal alsó sarokban kiszámításra kerül a napi makro tápanyag és kalória bevitelünk.

public int getOsszFeherje(string idopont)

{

List<EtkezesView> etkezesview = etkezesekviewn.FindAll(x => x.Idopont == idopont);

int totalfeherje = 0;

foreach(EtkezesView evv in etkezesview)

{

totalfeherje = totalfeherje + evv.Feherje;

}

return totalfeherje;

}

public int getOsszKaloria(string idopont)

{

List<EtkezesView> etkezesview = etkezesekviewn.FindAll(x => x.Idopont == idopont);

int totalkaloria = 0;

foreach (EtkezesView evv in etkezesview)

{

totalkaloria = totalkaloria + evv.Kaloria;

}

return totalkaloria;

}

Ezen a formon is elérhető a törlés funkció. Ha van kiválasztva sor, akkor felugrik egy ablak, amelyen megerősíthető a törlés.

### **2.2.3 Tesztelés**

A szoftverfejlesztés során elengedhetetlen az egységtesztek készítése. Ezekkel igazolható, hogy a programunk jól működik és megfelel a specifikációnak, valamint eleget tesz a megrendelőnek. A teszttel nem csak az elkészített kódot lehet leellenőrizni, hanem fel is lehet használni, például input adatok ellenőrzése során. Az alkalmazásban nem mindenhez készült teszt vagy ellenőrzés, de az étel hozzáadása során a mentés gomb megnyomásakor ellenőrzésre kerülnek a bemeneti adatok. Ha valamilyen hiba lépne fel a program kivételt dob az adott problémát megjelölve és egy errorProvider segítségével jelzésre kerül az adott mező mellett. Az első betű ellenőrzésére mutatnék egy tesztet példaképpen:

[TestMethod()]

public void validateTestEnevFirstLetterUpperCase()

{

try

{

Etel etel = new Etel("Csirkemell",2,3,4,5,"3g");

if (etel.validate())

{}

}

catch (Exception e)

{

Assert.Fail("Az étel első betűje nagybetű mégis hibát dob");

}

}

## **2.3 Web alkalmazás**

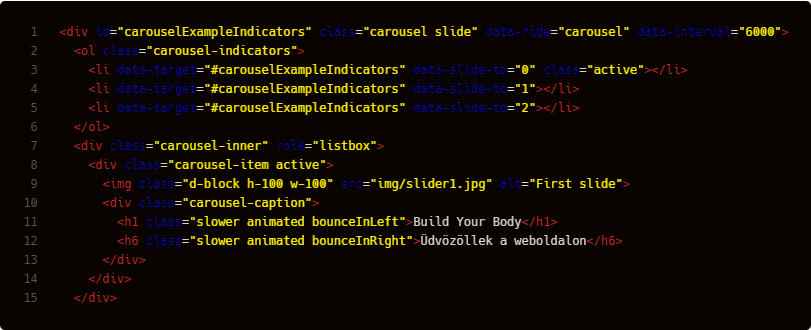
A web alkalmazás elkészítése során törekedtem arra, hogy a kezdő oldal minél átláthatóbb legyen, ezáltal nem használtam sok színt és kevés szöveget helyeztem el az oldalon. A fejlesztés során alkalmaztam programozási nyelveket és jelölőnyelvet egyaránt. Az oldal vázszerkezetét HTML jelölőnyelv segítségével készítettem el. Emellett a dizájnhoz igénybe vettem a Bootstrap 4- es verzióját, valamint CSS stíluslapokat is. A front-end oldalon használtam még jquery és javascript függvényeket.

Backend oldalon a PHP programozási nyelvet választottam. Ezt a nyelvet használtam a legtöbbet a tanulmányaim során, tehát nem is volt kérdés mivel dolgozzak. Ahhoz, hogy minél élvezhetőbbé tegyem a weboldalt igénybe vettem még az AJAX technológiát is. Ezáltal lehetővé válik a szerverrel való kapcsolattartás a teljes oldal újra töltése nélkül. Növelve a honlap sebességét és használhatóságát.

A fejlesztői környezet kiválasztása sem okozott problémát. Itt a Visual Studio Code nevezetű IDE-re esett a választás. Ugyanis számos olyan funkcióval rendelkezik, amelyek segítették a munkámat a kódolás során. Vegyük példaként a formatter funkcióját, amely egy gombkombinációra tagolja a több száz soros kódot. Ez a fejlesztés végéhez közeledve nagyon sokat segített, hogy minél jobban át lássam az általam megírt forráskódot.

### **2.3.1 Index**

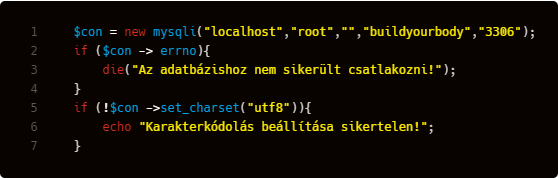
A kezdőoldalt másnéven indexnek nevezzük. Egy oldal betöltődésekor ez lesz az, amit automatikusan keresni fog a böngésző a fájlok között. A kezdőoldalt fontosnak tartottam, mert egy képet ad magáról az egész alkalmazásról. Amikor betöltődik az oldal a felhasználó egy Image slider-el találja magát szemben, amin elhelyeztem három darab idézetet. Az oldal tetején található még egy navigációs sáv, amelyen, ha kattintunk egy menüre oda scrollol a megfelelő tartalomhoz.



4. ábra Slider image

### **2.3.2 Csatlakozás az adatbázishoz**

Az adatbázishoz való csatlakozást a connect.php nevezetű fájlba oldottam meg, amely a php mappán belül található. Később becsatolásra kerül az init.php fájlba, amiben indításra kerül egy session, amely elengedhetetlen például a bejelentkezés során.



5. ábra Csatlakozás az adatbázishoz

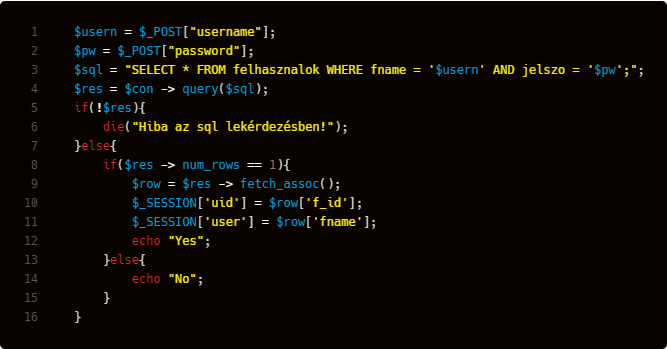
### **2.3.3 Bejelentkezés és regisztráció**

A regisztráció az egész rendszer alapja. Mivel az asztali alkalmazáson erre nincs lehetőség, így a weboldalon keresztül a látogatók fel tudnak regisztrálni és később a szoftverhez is hozzáférhetnek. Fontosnak tartom, hogy a regisztráció során a kliens valós és helyes adatokat adjon meg. Például az email cím formátum érvényessége. Ezek ellenőrzése megtörténik kliens oldalon jQuery és RegExp segítségével. Az adatokat utána AJAX-szal küldöm át szerver oldalra a reg.php nevű állományba, majd ezen keresztül viszem fel az adatbázisba.

A bejelentkezést úgy oldottam meg, hogy a menüsávban a belépés gombra kattintva egy modal ablak jelenik meg. Az első mezőbe a felhasználónév a másodikba a jelszó kerül. Ha helyesen van kitöltve, akkor egy AJAX kéréssel fut le, majd átirányításra kerül, az oldal a home.php-ra. Ha a bejelentkezés sikertelen nem töltődik újra az oldal, hanem a kliens oldalról hiba üzenetet jelenítek meg. Íme, néhány kódrészlet:



6. ábra Ajax kérés belépéskor

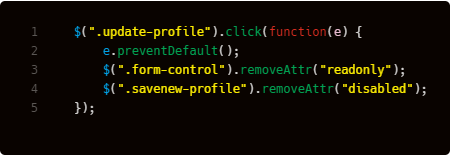


7. ábra Login.php

### **2.3.4 Profil**

A profil részben legelőször a kliens egy landing pagel találja magát szembe. Ezen az oldalon helyett kapott ugyanúgy felül egy navbar. Sikeres belépés után minden oldal kinézetéért egy másik stíluslap felel. Ez annyit takar, hogy az index külön stíluslappal van formázva. Az egész projektben két CSS fájlt használtam. Ennek oka, hogy ne legyen beömlesztve egy fájlba az összes stílus. Ezáltal könnyebben olvashatóbbak.

A bejelentkezett felhasználó adatai egy dropdown menüben kaptak helyet. Ez megtalálható minden php állományban a profile mappán belül. Fontos megemlítenem, hogy ezek az állományok bejelentkezés nélkül nem érhetőek el. Ugyanis, ha nincs a SESSION változóba semmi elmentve, akkor nem töltődnek be. Ez mindenhol vizsgálva van. A mentés gomb ki van kapcsolva, csak akkor érhető el, ha minden adat helyesen van kitöltve. A validáció során RegExp kifejezéseket használtam. Az input mezők readonly tulajdonságot kaptak és csak akkor módosítható a benne található érték, ha a Módosítás gombra kattintunk. Ekkor kliens oldalon lefut egy click esemény, amelyben eltávolításra kerül az attribútum a text mezőkről és a mentés gombról is.



8. ábra Readonly és disabled attribútum

Az edzéseket adatbázisból olvasom ki, majd egy táblázatban kerülnek eltárolásra. Minden sor végén elhelyezésre került egy trash icon, amely törlés funkciót hivatott szolgálni. Ezt kiválasztva AJAX kérésen belül szerver oldalra átküldésre kerül annak a sornak az ID-ja, majd szerver oldalon törlésre kerül az adatbázisból. Viszont az edzések csak akkor jelennek meg, ha az előtte lévő select elemből megtörténik az időpont kiválasztása. Az optionok között csak azok a dátumok találhatóak, amelyeknél a felhasználónak van edzésterve. Ezek után a html tagek között kiíratás kerül a select változó. Ez a php-n belül így néz ki:

$uid = $\_SESSION['uid'];

$sql = "SELECT DISTINCT idopont FROM edzesek WHERE f\_id ='$uid'";

$res = $con->query($sql);

$select = '<select class="custom-select date"><option>Időpontok</option>';

while ($row = $res->fetch\_array()) {

$select .= '<option id=' . $row[0] . '>' . $row[0] . '</option>';

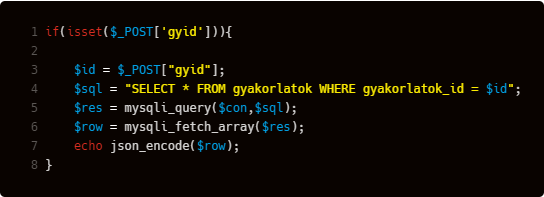
}

$select .= '</select>';

Új edzés hozzáadásakor egy modal ablak jelenik meg. Ahol két legördülő menüben jelenítem meg az izomcsoportokat és a hozzájuk tartozó gyakorlatokat. Hasonló elven, mint ahogy az fentebb látható a keltezéseknél. A gyakorlat menü akkor jelenik meg, ha az izomcsoport ki lett választva. A helyesség ellenőrzésére itt is sor került, mint minden formon. A gyakorlatok megtekintésére a megfelelő menüpontot kiválasztva van lehetősége a felhasználóknak. Itt úgy csináltam meg, hogy ne egy táblázatba kerüljön megjelenítésre az összes rekord. Így amellett döntöttem, hogy minden egyes gyakorlatot egy card elemben fogok megjeleníteni. Még egy pagination funkció is helyet kapott. Egyszerre csak 9 db rekord jelenik meg az oldalon és az alján elhelyezett lapozó elem segítségével lehet váltani a page-k között. A cardon belül látható a gyakorlat neve és egy rövid leírás a helyes kivitelezésről. Alulra pedig két oldalra egy trash és egy modify icon került. A kuka ikonra kattintva ID alapján törlődik az adatbázisból az adat oldal újratöltődés nélkül. A módosítás ikont kiválasztva egy modal ablakba jelennek meg a frissíteni kívánt elemek. Itt az AJAX kérésben adattípusként (dataType) Json formátumot alkalmaztam. A Json formátum kulcsértékpárokat tárol, és így volt a legegyszerűbb az új ablakba megjeleníteni az adatokat. Itt is lehetőség van új gyakorlat felvitelére ez ugyancsak felugró ablakba van kivitelezve. Az alábbi két ábra szemlélteti ezt:



9. ábra Ajax kérés json formátummal



10. ábra Json formátum visszaadása kliens oldalra

### **2.3.5 Reszponzivitás**

A mai technológiának köszönhetően a weboldalaknak szinte minden eszközön célszerű elérhetőnek lennie, így a web alkalmazások készítése során muszáj illeszkednie az oldalaknak a képernyő méretéhez. Mivel Bootstrap-et használtam, így segítségemre volt a keretrendszerben létrehozott rácsszerkezet, amelynek támogatásával könnyedén alakítottam ki reszponzív oldalakat. A külön használt stíluslapokon a media query részben csak olyan formázások vannak, mint például szöveg középre igazítása vagy margin beállítása.

## **2.4 Fejlesztési tervek**

Sajnos rengeteg olyan funkció nem szerepel a szoftverben, amely hasznos lenne. Ezek az idő hiánya miatt nem készültek el. Itt ezeket a funkciókat sorolom fel, hogy az alkalmazás egy következő életciklusba lépjen és egy kisebb célközönség használatba is vegye. A közeljövőben megvalósítandó tervek:

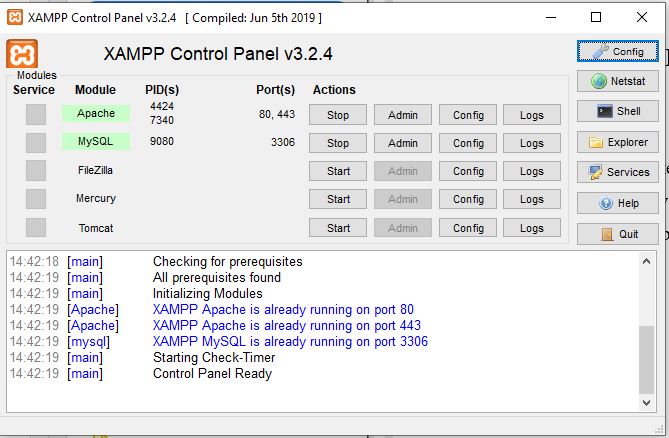
* Telefonos applikáció készítése
* Kommunikáció a felhasználók között (chat funkció)
* Szükséges makrotápanyagok kiszámítása testsúlykilógrammonként
* Személyi edzők funkció kiépítése (Étrend és edzésterv készítés saját vendég számára, vendégek kezelése).

# **3. Felhasználó Dokumentáció**

## **3.1 Asztali alkalmazás**

### **3.1.1 Program indítása**

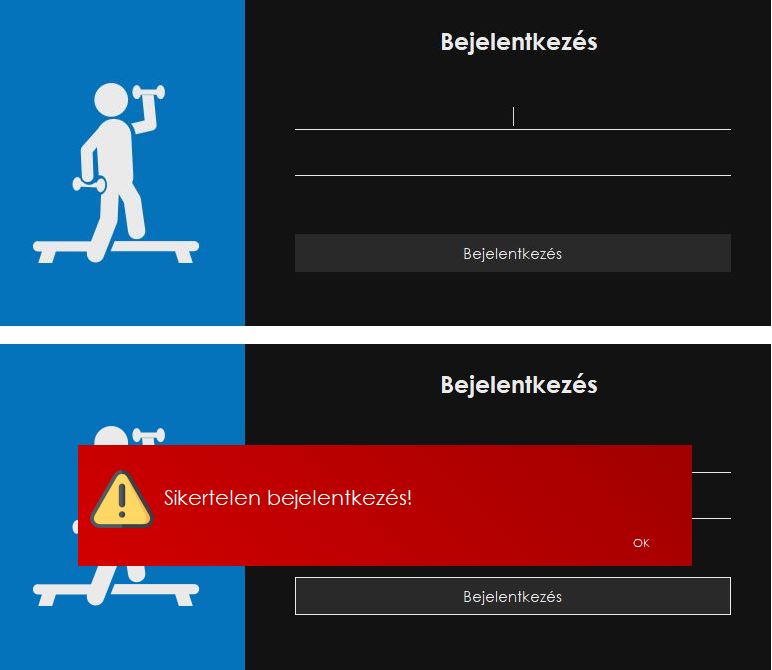
A programot a byb.exe-n keresztül lehet elérni. Mielőtt elindul, a program el kell indítani a XAMPP saját virtuális szervert, amelyen az adatbázis fut. Magát az adatbázist a program hozza létre, innentől már nincs több dolgunk vele. A XAMPP elindítása után be kell kapcsolni a MySql és Apache szolgáltatásokat.



11. ábra Xampp indítása Mysql és Apache szolgáltatások

### **3.1.2 Belépés a programba**

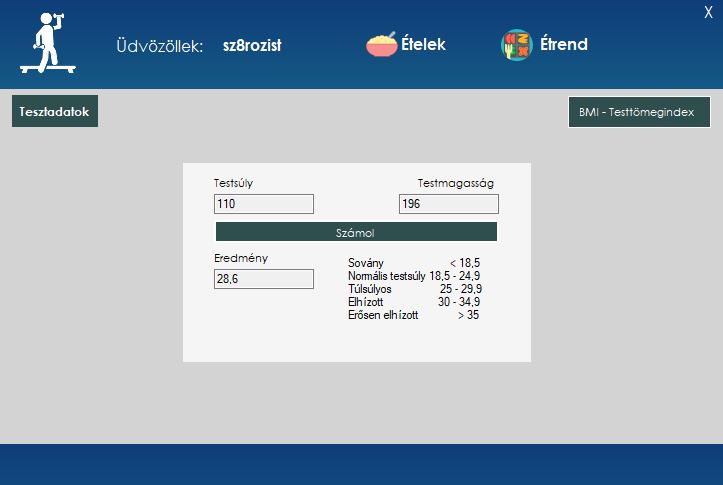
A program belépésénél csak regisztrált felhasználók tudnak megfelelő adatokat megadni. Indítás után felugrik egy bejelentkezési ablak, ahol a kliensnek a sikeres belépéshez érvényes felhasználónevet és jelszót kell megadni. Ha a sikeres a bejelentkezés, akkor a program tovább irányít az üdvözlő ablakba, ha sikertelen, akkor egy hibaüzenet jelenik meg a bejelentkezés ablakon, amely az üzenetet tartalmazza.



12. ábra Desktop bejelentkezés

### **3.1.3 Üdvözlő ablak**

A sikeres bejelentkezést követően a fő oldalra viszi a felhasználót a program. Itt található jelenleg egy Ételek és egy Étrend menüpont. Lehetőség van még testtömegindex kiszámítására, ezt a BMI – Testtömegindex gombra kattintva tudjuk elvégezni. Helyet kapott még egy tesztadatok gomb is, aminek a segítségével feltudjuk az adatbázisunkat tölteni adatokkal.



13. ábra Üdvözlő Ablak

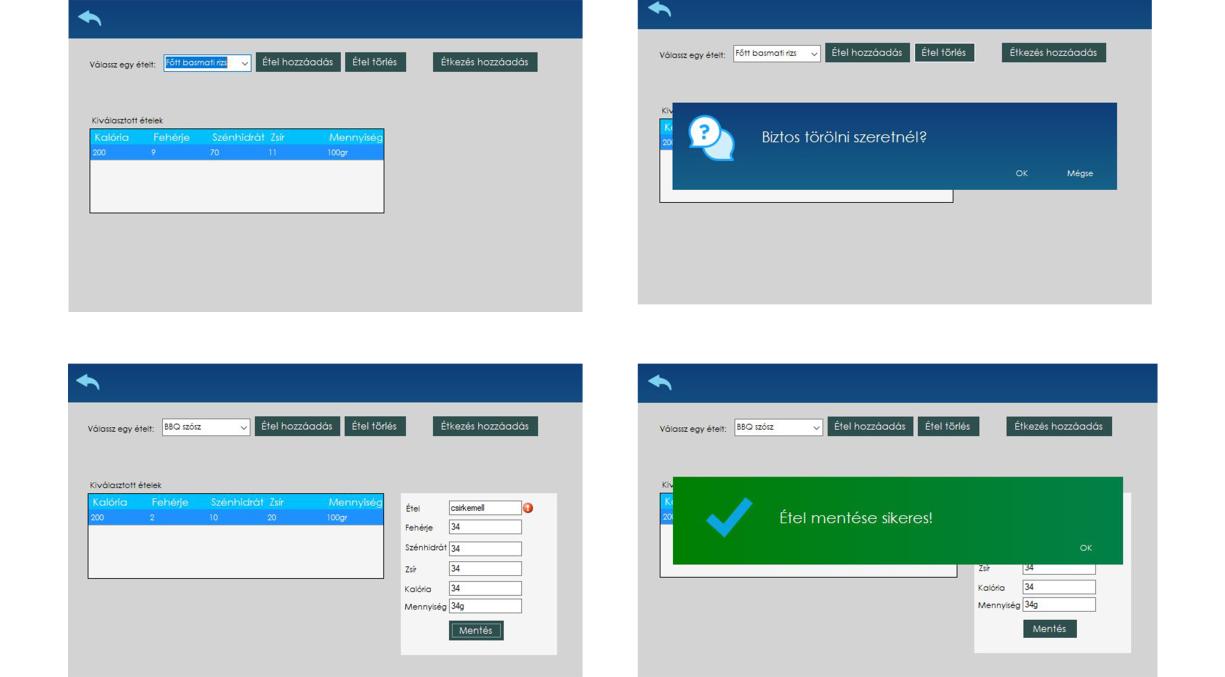
### **3.1.4 Ételek menüpont**

Ha a menüsávon az ételek gombra kattintunk, akkor felugrik egy új ablak. A menüsávban található egy vissza gomb, amit egy nyíl jelképez. A legördülő menüből ki tudjuk választani azt az ételt, amelynek látni szeretnénk a makrotápanyagait.

Új étel hozzáadása az étel hozzáadás gombbal lehetséges. A gomb megnyomásával egy új panel válik elérhetővé. Meg kell adnunk az étel nevét, fehérje tartalmát, szénhidrát tartalmát, zsír tartalmát, kalória tartalmát és a mennyiséget. A következő szempontokat kell szem előtt tartani az adatok megadásakor:

* Az étel nevének nagybetűvel kell kezdődnie.
* A fehérje, szénhidrát, zsír és kalória csak szám érték lehet.
* A mennyiségnél meg kell adni a mértékegységet a végén, tehát nem kezdődhet betűvel az adat.

Ha minden adat ki lett kitöltve, akkor a következő lépés a mentés gombra való kattintás. Ha minden adat megfelel a követelményeknek, akkor hozzáadásra kerül az új étel. Ha bármelyik szövegdobozban hibát talál, akkor azt egy hibaüzenettel jelzi a képernyőn a program.

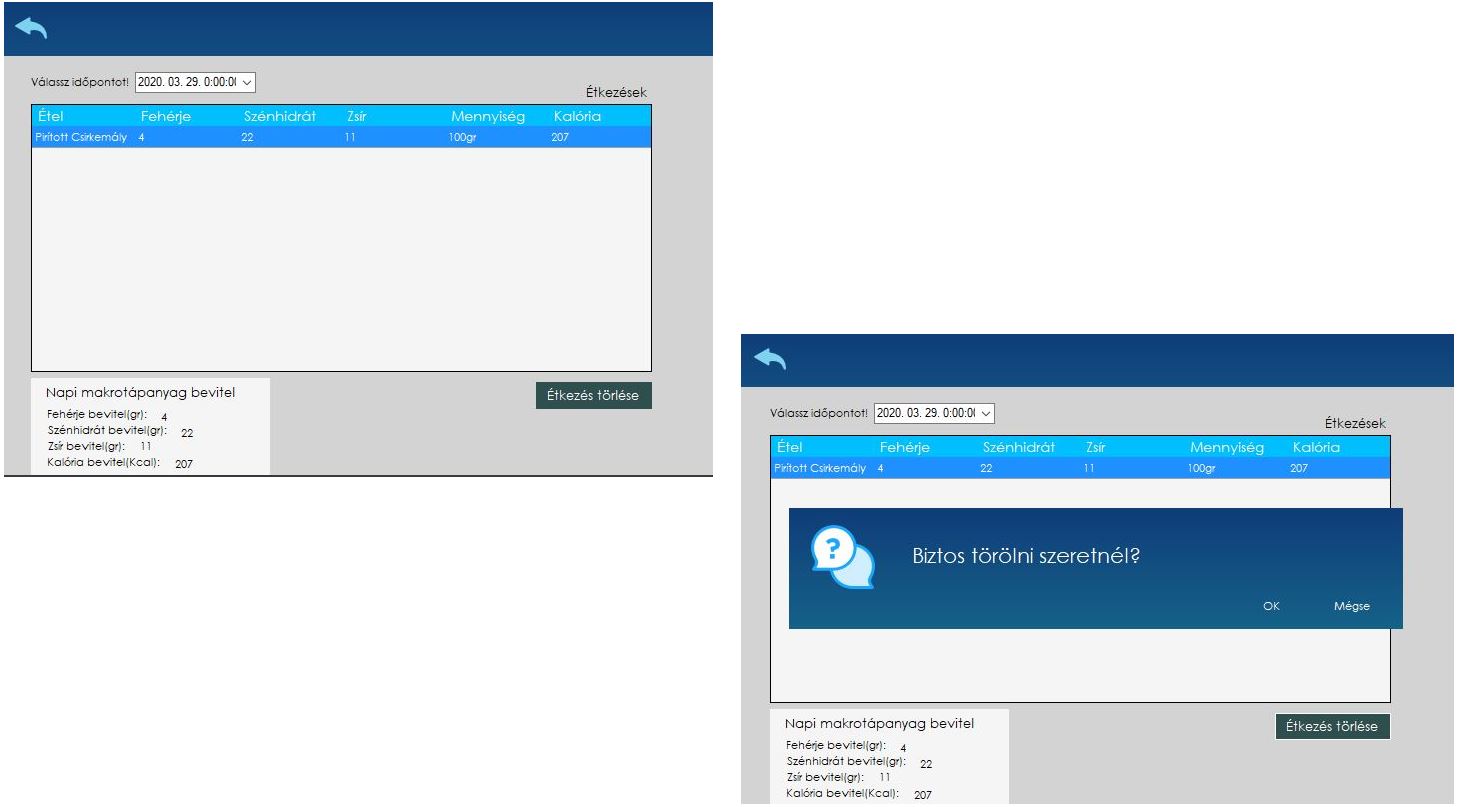


14. ábra Étel ablak

Az étkezés hozzáadás gombot kiválasztva ételt tudunk étkezéshez elmenteni. Ha rákattintunk, a gombra felugrik egy új panel, ahol ki kell, válasszuk az étkezés dátumát. Rákattintunk a mentés gombra és tárolásra kerül az étkezés.

### **3.1.5 Étrend menüpont**

Ha az étkezéseket szeretnénk megtekinteni, válasszuk ki az üdvözlő ablakon az étrend menüpontot a menüsávból. Kattintás után felugrik egy új ablak, amely az étkezéseinket tartalmazza. A legördülő menüben láthatjuk az étkezések időpontjait, amelyeknek a kiválasztása után megjelennek az adott napon elfogyasztott ételek. Az ablak bal alsó sarkában található egy napi makrotápanyag bevitel rész, ahol kiszámításra kerül az adott napon elfogyasztott fehérje, szénhidrát, zsír, kalória mennyisége. Van lehetőségünk étkezést eltávolítani. Ehhez válasszuk ki az étkezés törlése gombot. A program rákérdez, hogy biztos törölni szeretnénk-e. Ha igennel válaszolunk az alkalmazás törli a kiválasztott étkezést.



15. ábra Étrend menüpont

## **3.2 Web alkalmazás**

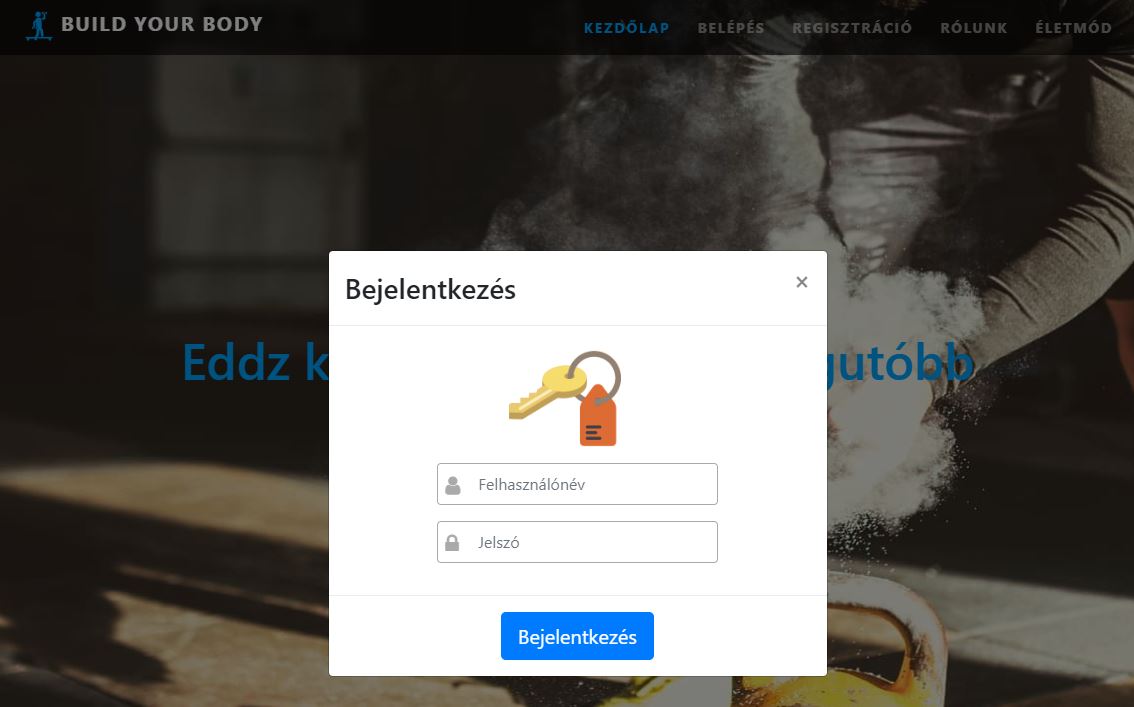
### **3.2.1 Oldalak**

A weboldalt igyekeztem úgy kialakítani, hogy az a felhasználói követelményeknek megfeleljen. A következő oldalak találhatóak az alkalmazáson belül:

* Kezdőoldal: Ez az első oldal, ami betöltésre kerül, amint megnyitja egy felhasználó az alkalmazást.
* Home: Ez az oldal csak bejelentkezés után érhető el.
* Gyakorlatok: Csak belépés után elérhető. Információs oldalként funkcionál a gyakorlatokról.
* Edzések: Az oldal csak bejelentkezés után érhető el. A saját edzések válnak elérhetővé. Melyik nap milyen gyakorlatokat végeztünk.

### **3.2.2 Funkciók**

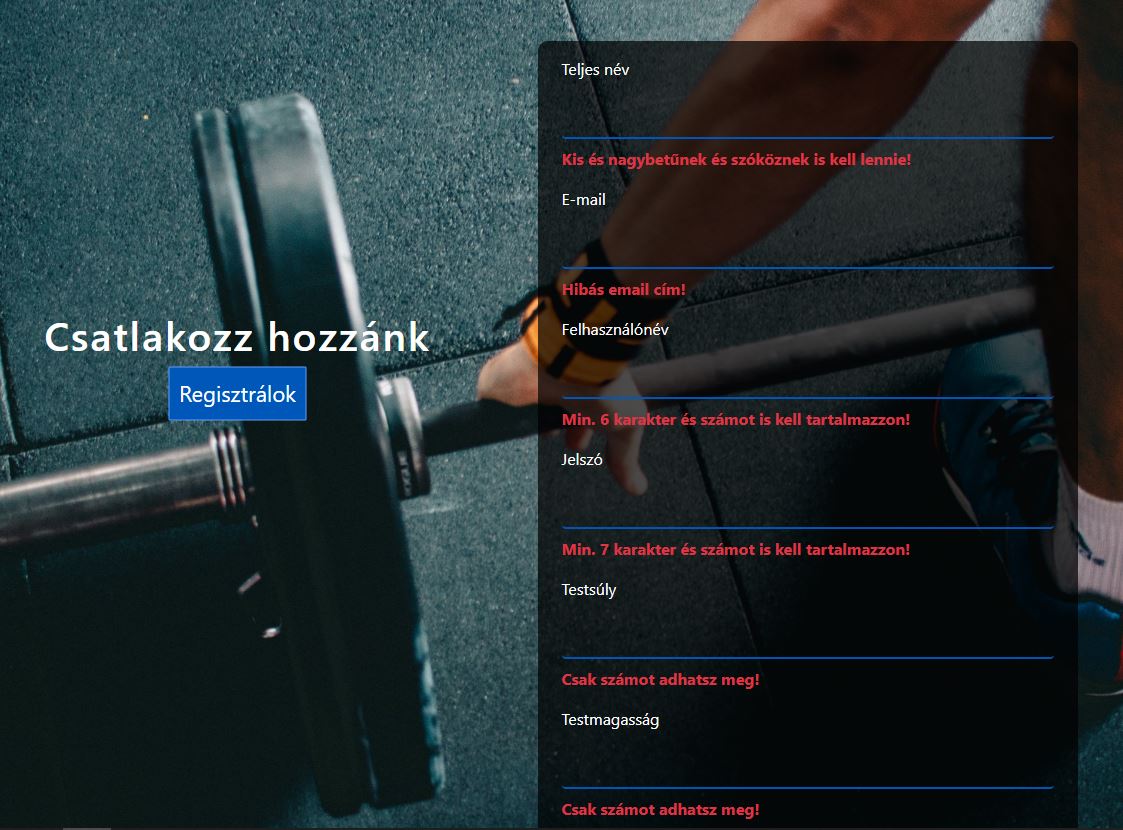
A bejelentkezés csak a regisztrált felhasználóknak adott. Szükség van hozzá egy felhasználónévre valamint egy jelszóra, amelyet a regisztrációkor adunk meg. Ha helyesen töltjük ki a két mezőt a belépő ablakba, akkor a háttérben ellenőrzésre kerül, hogy mindent megfelelően adtunk-e meg. Ezek után átirányításra kerülünk a home oldalra.



16. ábra Bejelentkezés

Regisztrációra a kezdőoldal alján van lehetőségünk. Ha rákattintunk a regisztráció menüpontra a navigációs sávon, akkor az oldal magától odavisz az űrlaphoz. Amint mindent helyesen kitöltünk, a rendszer visszaigazolja a sikeres regisztrációt. Ehhez viszont fontos odafigyelnünk a következőkre

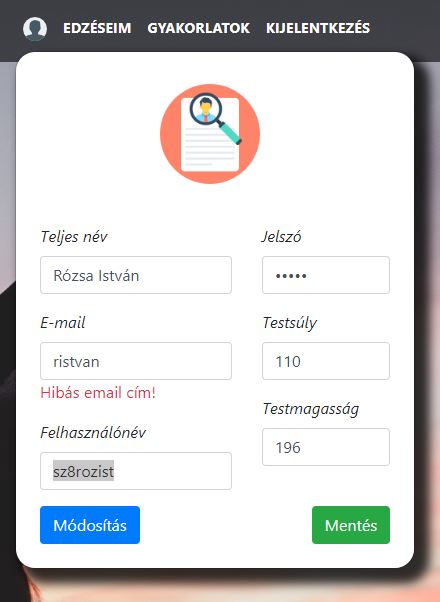
* A teljes névnek tartalmaznia kell kis és nagybetűt egyaránt, és lennie kell benne szóköznek.
* Az e-mail cím formájának megfelelőnek kell lennie. Ha nincs benne kukac vagy pont, akkor a rendszer nem fogja elfogadni azt.
* A felhasználónévnek legalább 6 karakter hosszúnak kell lennie és emellett számot is kell, hogy tartalmazzon.
* A jelszónak minimum 7 karakter hosszúnak kell lennie és kell, hogy legyen benne szám is.
* A testsúly és a testmagasság megadásakor törekedni kell arra, hogy a szöveges mezőbe betűk ne kerüljenek.



17. ábra Regisztráció

Ha valami mégis helytelenül kerül kitöltésre, ahogy azt az ábra is mutatja az alkalmazás minden esetben tájékoztatja a felhasználót.

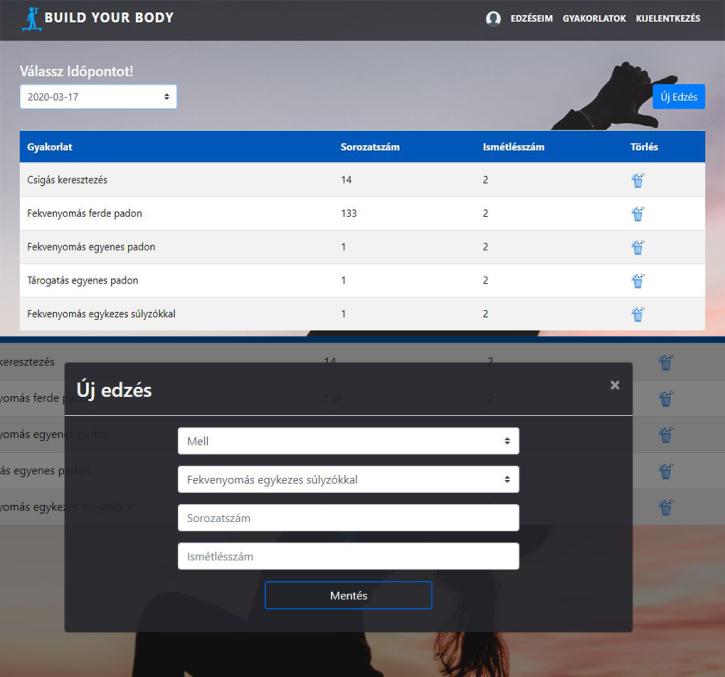
Saját adatok módosítására bejelentkezés után van lehetőség. A menüsávban a megfelelő menüpont kiválasztásával tehetjük ezt meg. Ha rákattintottunk a megfelelő gombra, akkor egy úgynevezett lenyíló menüben megjelenik előttünk az az űrlap, ahol a saját adataink tárolásra kerültek. Ha rákattintunk a módosítás gombra, akkor az összes szöveges mezőbe írási engedélyt kapunk és megváltoztathatjuk az adatainkat. Fontos megjegyezni, hogy az ellenőrzés során ugyanazok a szabályok vonatkoznak erre az űrlapra is, mint a regisztrációnál. Tehát ha valamit nem jól adunk meg, akkor hibaüzenet fog megjelenni. Ha helyesen van megadva minden adat, akkor a mentés gombra kattintva sikeres volt a frissítés.



18. ábra Saját adatok

### **3.2.2. Menüpontok**

Az edzéseim menüpont csak bejelentkezés után válik elérhetővé a felhasználó számára. Ezen az oldalon, nyomon követhetjük, hogy melyik nap milyen gyakorlatokat végeztünk. A legördülő menüből tudjuk kiválasztani a napot. Ha kiválasztásra került a nap, akkor megjelennek azok a gyakorlatok, amelyek ahhoz az időponthoz tartoznak. A kuka ikonnal tudunk törlést végrehajtani egyes gyakorlatokon. Valamint ha az új edzés gombra kattintunk, felugrik egy új ablak, ahol edzést tudunk elmenteni mindig az adott napra. Legördülő mezőből ki kell választanunk először az izomcsoportot, majd pedig magát a gyakorlatot. Utána meg kell, adjuk a sorozatszámot és az ismétlésszámot. Azonban a sorozatszám és ismétlésszám mezőbe csak számot írhatunk be. A program jelez, ha hibát észlel. Ha mindent helyesen adtunk meg, akkor sikeres lesz a mentés.



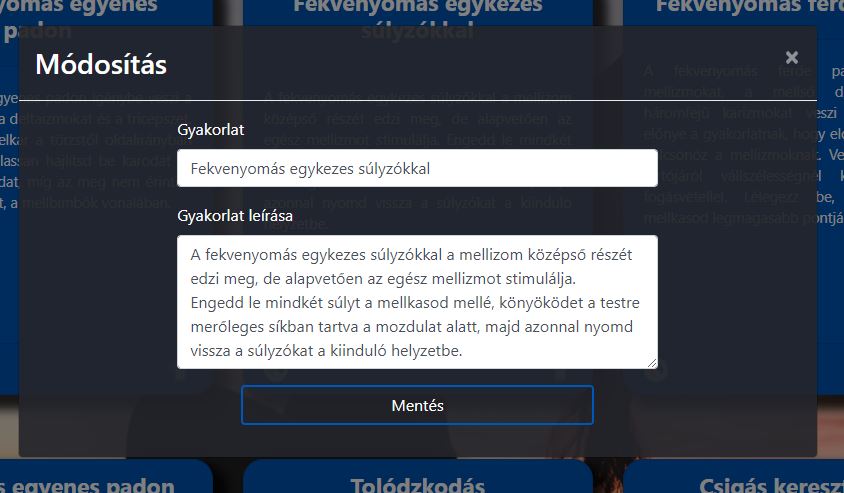
19. ábra Edzésim menüpont, Edzés hozzáadás

A gyakorlatok oldalon izomcsoportonként megtekinthetők az egyes gyakorlatok. Az izomcsoportot egy legördülő menüből kell kiválasztani. Kiválasztás után kék színű kis kártyákba jelennek meg a gyakorlatok és a hozzájuk tartozó leírások. A kártyák alján elhelyezésre került két kis ikon. A kuka ikonra kattintva ki tudjuk törölni az adott gyakorlatot.



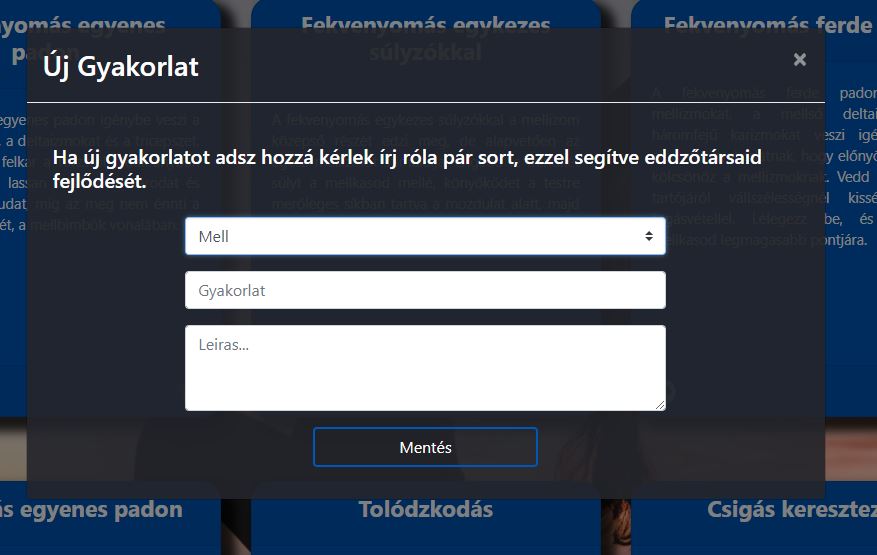
20. ábra Gyakorlatok oldal

A frissítés ikonra kattintva egy új ablakban tudjuk megváltoztatni az adott gyakorlatot. Figyelnünk kell arra, hogy a gyakorlat neve kizárólag nagybetűvel kezdődhez és egyik mezőt sem hagyhatjuk üresen.



21. ábra Gyakorlat módosítása

Az új gyakorlat gombra kattintva megjelenik egy másik ablak, amin legördülő sávból tudjuk kiválasztani az izomcsoportot, hogy melyikhez szeretnénk gyakorlatot hozzáadni. Meg kell adnunk a gyakorlat nevét. Ügyelnünk kell arra, hogy nagybetűvel kezdődjön a név, majd magáról a gyakorlatról kell írnunk pár sort ezzel segítve a többi edzőtársunkat. Legyen az akár haladó vagy kezdő.



22. ábra Új gyakorlat felvitele

1. 1 testtömeg-index [↑](#footnote-ref-1)