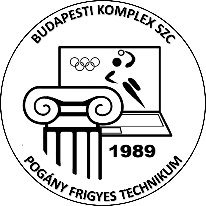
**BUDAPESTI KOMPLEX SZAKKÉPZÉSI CENTRUM**

**Pogány Frigyes Technikum**

**OM: 203032/012**

1183. Budapest, Thököly u. 11.

Tel.: 290-0642; e-mail: info@poganyszki.hu

**Ügyfélnyilvántartó alkalmazás**

**fitness-terem részére**

**Vizsgaremek**

**2022.**

**Készítette: Konzulens:**

Bagi Ferenc Tóth József

Hakkel Hedvig

Petrich Balázs Ferenc

Tartalomjegyzék

[A feladat leírása 3](#_Toc101724863)

[GitHub repository 3](#_Toc101724864)

[Az adatbázis elkészítése 4](#_Toc101724865)

[API elkészítése 9](#_Toc101724866)

[Az adminisztrátor felületének leírása 15](#_Toc101724867)

[A bérletszerkesztő (tulajdonosi) felület leírása 20](#_Toc101724868)

[Az ügyfél által megtekinthető felület leírása 23](#_Toc101724869)

[Az adatok küldésének és lekérésének tesztelése 26](#_Toc101724870)

[Az edzőterem kliens oldalának elkészítése 29](#_Toc101724871)

[Zárógondolatok a fenti ügyfélnyilvántartó program elkészítésének folyamatáról 35](#_Toc101724872)

# A feladat leírása

Fitness-termünk folyamatosan bővülő ügyfélköre a terem látogatottságának nyomon követhetőségét és az ügyféladatok pontos nyilvántartását teszi szükségessé részben marketing szempontból, részben a szolgáltatások kihasználtságának növelése céljából.

Feladatunk egy olyan nyilvántartó rendszer létrehozása, amelynek segítségével az ügyfélszolgálatos munkatárs képes új ügyfelet felvenni, adatait rögzíteni, módosítani és lekérdezni a náluk edző ügyfelek adatait, látogatásait, továbbá rögzítheti a munkaideje alatt értékesített belépőket, valamint ellenőrizheti a bérletek érvényességét, s a még hátralévő szabad alkalmak mennyiségét.

A feltöltött adatok online adatbázisban kerülnek tárolásra, amelyet – meghatározott jogosultsággal – a tulajdonos és az ügyfelek is elérhetnek egy API-n keresztül.

Az ügyfelek csak a saját adataikhoz férhetnek hozzá, míg a tulajdonos jegy, és bérlettípusokat tud létrehozni, módosítani és törölni, ami az eladott bérletek korábban rögzített adatait már nem befolyásolja.

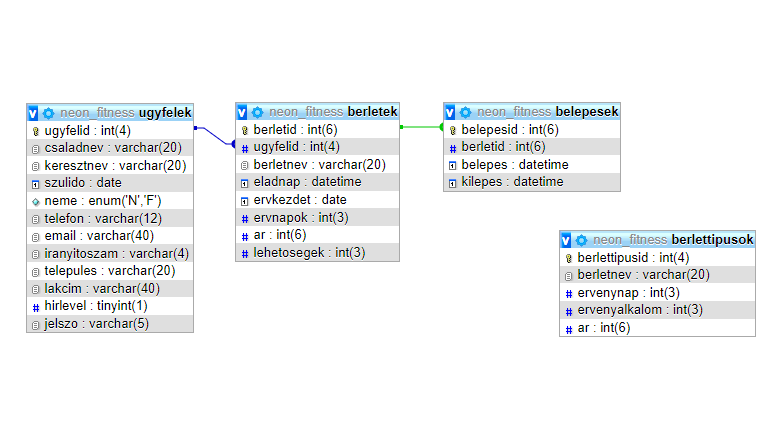
# GitHub repository

Az alkalmazások kódjai ebben a repository-ban érhetők el:

<https://github.com/PetrichBF/ugyfelnyilvantartas/>

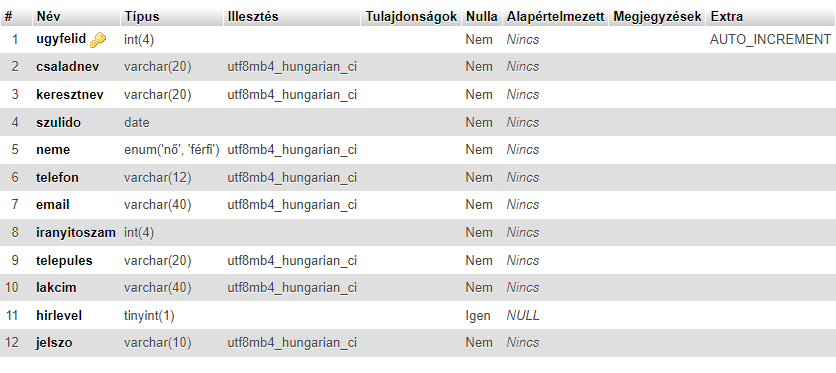
# Az adatbázis elkészítése

Az adatbázist adatbázis-kezelővel készítettük el. Az adatokat az alábbi négy táblára bontottuk:



**Ügyfelek tábla:**

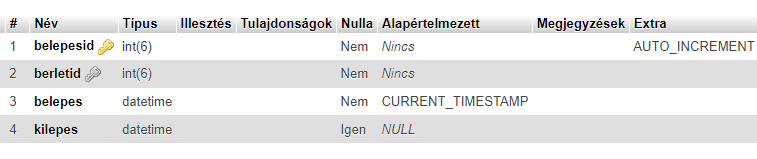
Ebben rögzítettük az ügyfelek adatait, valamint az általuk választott jelszót, a web és mobil applikációs felületre történő belépéshez. Elsődleges kulcsként az ügyfél azonosító került megjelölésre. Ennek alapján minden további adat az adott ügyfélhez lesz kapcsolható. A táblában a hírlevél igénye is rögzíthető opció.



| **mezőnév** | **típus** | **érték** | **leírás** | **mező indoklása** |
| --- | --- | --- | --- | --- |
| ugyfelid | INT(4) | auto increment | ügyfél azonosítója, ami egy automatikusan növekvő, szám típusú mező | ügyfelet egyértelműsíti  4 számjeggyel 10000 főt lehetne tárolni |
| csaladnev | VARCHAR(20) NOT NULL | tetszőleges szöveg | ügyfél családneve | azonosításhoz szükséges |
| keresztnev | VARCHAR(20) NOT NULL | tetszőleges szöveg | ügyfél keresztnevei | azonosításhoz szükséges |
| szulido | DATE NOT NULL | (elmúlt) dátum | ügyfél születési ideje | azonosításhoz szükséges |
| neme | ENUM NOT NULL | F, N | ügyfél neme rövidítve | a továbbfejlesztés során a statisztikához, az automatikus öltözőkulcs kiadáshoz szükséges |
| telefon | VARCHAR(12) NOT NULL | (telefon)szám szövegesen | ügyfél telefonszáma | kapcsolattartáshoz (ügyfél értesítéséhez) szükséges adat  +36xxyyyyyyy 12 karakter |
| email | VARCHAR(50) NOT NULL | teszőleges szöveg (e-mail cím, azaz @ és . tartalom vizsgálatra kerül) | ügyfél e-mail címe | kapcsolattartáshoz (ügyfél tájékoztatásához) szükséges adat |
| iranyitoszam | VARCHAR(4) NOT NULL | szövegként tárolt szám | ügyfél lakcímének irányítószáma (feltételezve, hogy Magyarországon lakik, ezért csak négy karakter hosszban) | kapcsolattartáshoz |
| telepules | VARCHAR(20) NOT NULL | szöveg | ügyfél lakcíméhez tartozó település | kapcsolattartáshoz (fejlesztés során külön tábla alkalmazása alapján az adott irányítószámhoz meghatározható a település neve is) |
| lakcim | VARCHAR(40) NOT NULL | szöveg | ügyfél utca, házszám, emelet, ajtó adata összevontan | kapcsolattartáshoz (nem tűnik jelenleg indokoltnak ezt további mezőkre bontani) |
| hirlevel | BOOLEAN NOT NULL | igen, nem | ügyfél kér-e hírlevelet | nem feltétlen indokolt, de tudjuk majd vele változtatni |
| jelszo | VARCHAR(64) NOT NULL | (titkosított) jelszó szövegmező | ügyfél jelszavának hash kódja a saját adatai megtekintéséhez, egyes mezők változtatásához | ellenőrizni kell, hogy valóban az ügyfél nézi meg az adatait |

**Belépések tábla:**

A táblában rögzíthető adatok az ügyfélforgalmat - ezen belül a napijeggyel belépők számát -, valamint a létesítmény napszak szerinti látogatottságát segítik nyomon követni. A 3. és 4. sorok adatai alapján a belépések és kilépések pontos ideje is rögzítésre kerül.



| **mezőnév** | **típus** | **érték** | **leírás** | **mező indoklása** |
| --- | --- | --- | --- | --- |
| belepesid | INT(6) | auto increment | belépés azonosítója, ami egy automatikusan növekvő, szám típusú mező |  |
| berletid | INT(6) NOT NULL |  | melyik bérlettel történt belépés |  |
| belepes | DATETIME | most | mikor jött |  |
| kilepes | DATETIME | most | mikor távozott |  |

**Bérletek tábla:**

Az ügyfeleknek eladott bérlet-adatok nyilvántartására szolgál, egyben lehetővé teszi az egyes bérletek konkrét ügyfelekhez való kapcsolását (1-3 sorok: berletid, ugyfelid, berletnev), továbbá a lejárati idő, és a felhasznált alkalmak nyomon követését is. (5-6 és 8. sor)

A tábla “ar” mezője biztosítja a bérletárak mindenkori aktualizálását, ezért, csak a tulajdonos által szerkeszthető.



| **mezőnév** | **típus** | **érték** | **leírás** | **mező indoklása** |
| --- | --- | --- | --- | --- |
| berletid | INT(6) | auto increment | belépés, illetve lehetőség azonosítója, ami egy automatikusan növekvő, szám típusú mező | azonosító mező |
| ugyfelid | INT(4) NOT NULL |  | ügyfél azonosítója | meghatározni, hogy ki lép be |
| berletnev | VARCHAR(20) NOT NULL |  | bérletnév | meghatározni a bérlet nevét (könnyebb erre hivatkozni, mint egy számra) |
| eladnap | DATE NOT NULL | dátum (mai nap) | mikor került eladásra a bérlet | statisztikához (adott időszakban mi lett eladva) |
| ervkezdet | DATE NOT NULL | dátum (mai vagy jövőbeli) | mikortól érvényes a bérlet | elővételbe is meg lehet venni |
| ervvege | DATE NOT NULL | dátum (mai vagy jövőbeli) | meddig érvényes a bérlet |  |
| lehetosegek | INT(3) | szám (999-0) | még hány belépésre van lehetőség (0 – elfogyott) | itt számolok vissza, kivéve 999-ből, mert azt nem csökkentem |
| ar | INT(6) NOT NULL | 0-999999 közötti szám | a bérlet vételkori ára | statisztikához – 6 karakter indokolt, életszerű a 100000 Ft feletti ár éves bérlet esetében; esetleg lehettet árváltozás is a vásárlás óta, ezért kerül itt rögzítésre az ár |

**Bérlettípusok tábla:**

Közvetlenül egyik táblához sem kapcsolódik, mivel adatai révén a különböző bérlettípusok, azok érvényessége és a még felhasználható alkalmak mennyisége rögzíthető. Mindezek által háttér információként szolgálnak a terem működtetőjének, módosításuk csak a tulajdonos számára megengedett.



| **mezőnév** | **típus** | **érték** | **leírás** | **mező indoklása** |
| --- | --- | --- | --- | --- |
| berlettipusid | INT(4) | auto increment | bérlettípus azonosítója, ami egy automatikusan növekvő, szám típusú mező | bérlettípus meghatározása, amivel a belepesek táblában az ügyfélhez hozzárendelem  4 számjeggyel eléggé túl van biztosítva |
| berletnev | VARCHAR(20) | szöveg | bérlet neve (napijegy, 30 napos, féléves, jóga, pilates, stb..) | egyszerűbb így hivatkozni rá |
| ervenynap | INT(3) NOT NULL | 1 és 365 közé eső szám | bérlettípus maximális érvénytartama | meghatározni, hogy az érvényesség kezdetétől hány napig lehet felhasználni  999 napig lehetne elmenni, 3 számjegy mindenképpen szükséges, több nem indokolt |
| ervenyalkalom | INT(3) NOT NULL | 1 és 999 közé eső szám | hány belépésre jogosít (1 a napijegy, x az x alkalmas, 999 a korlátlan) | meghatározni, hogy az érvényesség idején belül hány alkalommal léphet be |
| ar | INT(6) NOT NULL | 0-999999 közötti szám | a bérlet eladási ára | értékesítéshez – 6 karakter indokolt, életszerű a 100000 Ft feletti ár éves bérlet esetében |

# API elkészítése

Az App.js fájl végpontjainak táblázatos leírása

|  |  |
| --- | --- |
| **Végpont:** | /ugyfelek |
| Metódus: | GET |
| SQL lekérdezés: | SELECT \* FROM ugyfelek |
| Feladat: | ügyfelek adatainak lekérdezése |
| Felhasználás: | * adminisztrátor oldalán az ügyféllista megjelenítése |
| Metódus: | POST |
| SQL lekérdezés: | INSERT INTO ugyfelek (csaladnev, keresztnev, szulido, neme, telefon, email, iranyitoszam, telepules, lakcim, hirlevel, jelszo) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?) |
| Feladat: | új ügyfél adatainak rögzítése |
| Felhasználás: | * az adminisztrátor oldalán új ügyfél adatainak rögzítése |
| Metódus: | PATCH |
| SQL lekérdezés: | UPDATE ugyfelek SET csaladnev = ?, keresztnev = ?, szulido = ?, neme = ?, telefon = ?, email = ?, iranyitoszam = ?, telepules = ?, lakcim = ?, hirlevel = ?, jelszo = ? WHERE ugyfelid = ? |
| Feladat: | meglévő ügyfél adatainak módosítása, beleértve a jelszómezőt is, ezáltal megvalósítja az elfelejtett jelszó helyett egy új jelszó beállítását |
| Felhasználás: | * az adminisztrátor oldalán a már meglévő ügyfél adatainak módosítása és/vagy egy új jelszó beállítása |

|  |  |
| --- | --- |
| **Végpont:** | /ugyfelekjn |
| Metódus: | PATCH |
| SQL lekérdezés: | UPDATE ugyfelek SET csaladnev = ?, keresztnev = ?, szulido = ?, neme = ?, telefon = ?, email = ?, iranyitoszam = ?, telepules = ?, lakcim = ?, hirlevel = ? WHERE ugyfelid = ? |
| Feladat: | meglévő ügyfél adatainak módosítása a jelszómező nélkül |
| Felhasználás: | * az adminisztrátor oldalán a már meglévő ügyfél adatainak módosítása |

|  |  |
| --- | --- |
| **Végpont:** | /ugyfelek/:ugyfelid |
| Metódus: | GET |
| SQL lekérdezés: | SELECT \* FROM ugyfelek WHERE ugyfelid = ? |
| Feladat: | egy adott azonosítójú ügyfél adatainak lekérdezése |
| Felhasználás: | * az adminisztrátor oldalán egy meglévő ügyfél adatainak lekérdezése |

|  |  |
| --- | --- |
| **Végpont:** | /berlettipusok |
| Metódus: | GET |
| SQL lekérdezés: | SELECT \* FROM berlettipusok |
| Feladat: | a napijegyek, bérlettípusok adatainak lekérdezése |
| Felhasználás: | * bérleteket rögzítő felületen (a tulajdonos részére) történő megjelenítés * adminisztrátor felületén történő megjelenítés * a terem oldalán (ügyféloldal) történő megjelenítés |
| Metódus: | POST |
| SQL lekérdezés: | INSERT INTO berlettipusok (berletnev, ervenynap, ervenyalkalom, ar) VALUES (?, ?, ?, ?) |
| Feladat: | új napijegy, bérlet rögzítése |
| Felhasználás: | * bérleteket rögzítő felületen (a tulajdonos részéről) történő új rögzítés |
| Metódus: | PATCH |
| SQL lekérdezés: | UPDATE berlettipusok SET berletnev = ?, ervenynap = ?, ervenyalkalom = ?, ar = ? WHERE berlettipusid = ? |
| Feladat: | adott azonosítójú napijegy, bérlet adatainak módosítása |
| Felhasználás: | * bérleteket rögzítő felületen (a tulajdonos részéről) történő módosítás |

|  |  |
| --- | --- |
| **Végpont:** | /berlettipusok/:berlettipusid |
| Metódus: | GET |
| SQL lekérdezés: | SELECT \* FROM berlettipusok WHERE berlettipusid = ? |
| Feladat: | adott azonosítójú napijegy, bérlet adatainak lekérése |
| Felhasználás: | * bérleteket rögzítő felületen (a tulajdonos részére) történő bérletadatok megjelenítése |
| Metódus: | DELETE |
| SQL lekérdezés: | DELETE FROM berlettipusok WHERE berlettipusid = ? |
| Feladat: | adott azonosítójú napijegy, bérlet törlése |
| Felhasználás: | * bérleteket rögzítő felületen (a tulajdonos részéről) történő törlés, amely nincs kihatással a már eladott bérletek adataira |

|  |  |
| --- | --- |
| **Végpont:** | /ervenyesberletek |
| Metódus: | GET |
| SQL lekérdezés: | SELECT berletek.\* , DATE\_ADD(ervkezdet, INTERVAL ervnapok - 1 DAY) as ervvege, ugyfelek.keresztnev, ugyfelek.csaladnev FROM berletek, ugyfelek WHERE (lehetosegek > 0) && (ervkezdet <= now()) && (DATE\_ADD(ervkezdet, INTERVAL ervnapok DAY) > now()) && (berletek.ugyfelid = ugyfelek.ugyfelid) |
| Feladat: | a lekérdezés napján érvényes bérletek lekérdezése |
| Felhasználás: | * az adminisztrátor felületén a beléptetéshez összeállított lista * a tulajdonos felületén az érvényes bérletekről történő tájékoztatás |

|  |  |
| --- | --- |
| **Végpont:** | /ervenyesberletek/:ugyfelid |
| Metódus: | GET |
| SQL lekérdezés: | SELECT \* , DATE\_ADD(ervkezdet, INTERVAL ervnapok - 1 DAY) as ervvege FROM berletek WHERE (lehetosegek > 0) && (ervkezdet <= now()) && (DATE\_ADD(ervkezdet, INTERVAL ervnapok DAY) > now()) && ugyfelid = ? |
| Feladat: | egy adott azonosítójú ügyfél érvényes bérletadatainak lekérése |
| Felhasználás: | * adminisztrátor felületén az adott ügyfél kiválasztása után az érvényes bérleteinek és beléptetéshez történő megjelenítése |

|  |  |
| --- | --- |
| **Végpont:** | /berletek/:ugyfelid |
| Metódus: | GET |
| SQL lekérdezés: | SELECT \* , DATE\_ADD(ervkezdet, INTERVAL ervnapok - 1 DAY) as ervvege FROM berletek WHERE ugyfelid = ? |
| Feladat: | egy adott azonosítójú ügyfél valamennyi bérletadatainak lekérése |
| Felhasználás: | * adminisztrátor felületén az adott ügyfél kiválasztása után valamennyi bérleténadatainak megjelenítése, továbbá az érvényes bérletek beléptetéshez történő megjelenítése |

|  |  |
| --- | --- |
| **Végpont:** | /berletek |
| Metódus: | GET |
| SQL lekérdezés: | SELECT \* , DATE\_ADD(ervkezdet, INTERVAL ervnapok - 1 DAY) as ervvege FROM berletek, ugyfelek WHERE berletek.ugyfelid = ugyfelek.ugyfelid |
| Feladat: | valamennyi bérlet adatainak lekérdezése |
| Felhasználás: | * az adminisztrátor felületén az eladott bérletekről összeállított lista * a tulajdonos felületén az eladott bérletekről történő tájékoztatás |
| Metódus: | POST |
| SQL lekérdezés: | INSERT INTO berletek (ugyfelid, berletnev, eladnap, ervkezdet, ervnapok, ar, lehetosegek) VALUES (?, ?, now(), ?, ?, ?, ?) |
| Feladat: | az ügyfél részére eladott bérlet adatainak rögzítése |
| Felhasználás: | * az adminisztrátor felületén történő bérleteladáskor |
| Metódus: | PATCH |
| SQL lekérdezés: | UPDATE berletek SET lehetosegek = (lehetosegek - 1) WHERE berletid = ? && lehetosegek < 999 |
| Feladat: | az adott bérlet esetén a további alkalmak csökkentése, ha az ügyfél belépett vele és nem korlátlan belépésre jogosított |
| Felhasználás: | * az adminisztrátor felületén történő beléptetés rögzítésekor |

|  |  |
| --- | --- |
| **Végpont:** | /belepesek |
| Metódus: | GET |
| SQL lekérdezés: | SELECT belepesek.\*, ugyfelek.csaladnev, ugyfelek.keresztnev FROM belepesek, ugyfelek, berletek WHERE belepesek.berletid = berletek.berletid AND berletek.ugyfelid = ugyfelek.ugyfelid ORDER BY belepesek.belepes DESC |
| Feladat: | fordított időrendi sorrendben a belépések lekérdezése |
| Felhasználás: |  |
| Metódus: | POST |
| SQL lekérdezés: | INSERT INTO belepesek (berletid, belepes) VALUES (?, now()) |
| Feladat: | a belépések táblában az ügyfél adott bérlettel történő belépésének idejének rögzítése |
| Felhasználás: | * az adminisztrátor felületén a beléptetés rögzítése |
| Metódus: | PATCH |
| SQL lekérdezés: | UPDATE belepesek SET kilepes = now() WHERE belepesid = ? |
| Feladat: | a belépések táblában az ügyfél kilépésének idejének rögzítése |
| Felhasználás: | * az adminisztrátor felületén a kiléptetés rögzítése |

|  |  |
| --- | --- |
| **Végpont:** | /kileptetendo |
| Metódus: | GET |
| SQL lekérdezés: | SELECT belepesek.\*, ugyfelek.csaladnev, ugyfelek.keresztnev FROM belepesek, ugyfelek, berletek WHERE kilepes is NULL && belepesek.berletid = berletek.berletid AND berletek.ugyfelid = ugyfelek.ugyfelid |
| Feladat: | a jelenleg bent lévő ügyfelek listájának összeállítása a kiléptetéshez |
| Felhasználás: | * adminisztrátor felületén az ügyfél kiléptetése |

|  |  |
| --- | --- |
| **Végpont:** | /kereses |
| Metódus: | GET |
| SQL lekérdezés: | SELECT \* FROM ugyfelek WHERE CONCAT(csaladnev,' ', keresztnev) LIKE '%?%' |
| Feladat: | az ügyfél keresése a név részlete alapján |
| Felhasználás: | * adminisztrátor felületén egy adott ügyfél megkeresése a neve vagy annak részlete alapján |

**Ügyfél bejelentkezése:**

|  |  |
| --- | --- |
| **Végpont:** | /users/login |
| Metódus: | POST |
| SQL lekérdezés: | SELECT \* FROM ugyfelek WHERE ugyfelid = ? |
| Feladat: | az ügyfél bejelentkezésékor a jogosultság megállapítása |
| Felhasználás: | * ügyféloldalon történő bejelentkezés során a jogosultság ellenőrzése és a token küldése |

**Ügyfél részéről végrehajtható lekérdezések a bejelentkezés után:**

|  |  |
| --- | --- |
| **Végpont:** | /azonositottugyfel/:ugyfelid |
| Metódus: | GET |
| SQL lekérdezés: | SELECT \* FROM ugyfelek WHERE ugyfelid = ? |
| Feladat: | a bejelentkezett ügyfél (saját) adatainak lekérése |
| Felhasználás: | * ügyféloldalon bejelentkezés után a személyes adatok megjelenítéséhez adatok küldése |

|  |  |
| --- | --- |
| **Végpont:** | /azonositottugyfelbelepesek/:ugyfelid |
| Metódus: | GET |
| SQL lekérdezés: | SELECT belepesek.\* FROM belepesek, berletek WHERE belepesek.berletid = berletek.berletid AND berletek.ugyfelid = ? ORDER BY belepesek.belepes DESC |
| Feladat: | a bejelentkezett ügyfél (saját) belépéseinek lekérése |
| Felhasználás: | * ügyféloldalon bejelentkezés után a korábbi belépési adatok megjelenítéséhez |

|  |  |
| --- | --- |
| **Végpont:** | /azonositottugyfelbelepesek5/:ugyfelid |
| Metódus: | GET |
| SQL lekérdezés: | SELECT belepesek.\* FROM belepesek, berletek WHERE belepesek.berletid = berletek.berletid AND berletek.ugyfelid = ? ORDER BY belepesek.belepes DESC LIMIT 5 |
| Feladat: | a bejelentkezett ügyfél (saját) utolsó 5 belépésének lekérése |
| Felhasználás: | * ügyféloldalon bejelentkezés után a legutolsó 5 belépés adatainak megjelenítéséhez |

|  |  |
| --- | --- |
| **Végpont:** | /azonositottervenyesberletek/:ugyfelid |
| Metódus: | GET |
| SQL lekérdezés: | SELECT \* , DATE\_ADD(ervkezdet, INTERVAL ervnapok - 1 DAY) as ervvege FROM berletek WHERE (lehetosegek > 0) && (ervkezdet <= now()) && (DATE\_ADD(ervkezdet, INTERVAL ervnapok DAY) > now()) && ugyfelid = ? |
| Feladat: | a bejelentkezett ügyfél (saját) érvényes bérleteinek lekérése |
| Felhasználás: | * ügyféloldalon bejelentkezés után az ügyfél érvényes bérleteinek adatai megjelenítéséhez |

|  |  |
| --- | --- |
| **Végpont:** | /azonositottberletek/:ugyfelid |
| Metódus: | GET |
| SQL lekérdezés: | SELECT \* , DATE\_ADD(ervkezdet, INTERVAL ervnapok - 1 DAY) as ervvege FROM berletek WHERE ugyfelid = ? |
| Feladat: | a bejelentkezett ügyfél (saját) bérleteinek lekérése |
| Felhasználás: | * ügyféloldalon bejelentkezés után az ügyfél összes bérletének adatai megjelenítéséhez |

**Az alkalmazás általános ismertetése**

Az alkalmazás három felületből áll, két felület csak a termen belül használatos, a harmadik felület az interneten keresztül elérhető az ügyfelek részére is. Ez utóbbi felület egyes részeinek megtekintése jogosultságtól függ, ezért az ügyfélnek az azonosítójával és a legalább 5 karakter hosszú jelszavával be kell jelentkeznie.

A tulajdonos és az adminisztrátor felülete – mivel ezek belső gépeken futnak – nem rendelkeznek külön jelszavas beléptetéssel.

Kezdetben a tulajdonosi felületen kell létrehozni a kínálatnak megfelelően a bérlet és jegy típusait, melyeket az adminisztrátori felületen ezt követően majd ki lehet választani.

Az adminisztrátornak az ügyfél adatait fel kell vinnie, ezt követően – az előzőleg rögzített bérletek közül – már értékesíthet az ügyfél részére. Amennyiben az ügyfélnek már van érvényes bérlete, a beléptetés is megvalósítható, valamint a bent lévő ügyfél ki is léptethető.

# Az adminisztrátor felületének leírása

Az „oldalFrissit” ismertetése:

Az eljárás négy további eljárást hív meg:

* berlettipusokLista,
* belepesekLista,
* ugyfelekLista,
* berletekLista.

Az „ugyfelFrissit” ismertetése:

Az eljárás két további eljárást hív meg:

* ugyfelAdatok,
* ugyfelBerletekLista.

Az „ugyfelekLista” ismertetése:

Feladata a szűrő tartalmának megfelelő ügyfelek adatainak legördülő menübe történő rendezése. Megjeleníti a találatok számát is.

A „berlettipusokLista” ismertetése:

Feladata két lista (berlettipusokLista és berlettipusokLista1) elkészítése, melyből az előző eredménye egy táblázatos formátumú megjelenítés és adatbázisban tárolt adatok felhasználásával az alábbi oszlopok szerint:

* ID – bérlettípus azonosítója
* Elnevezés – bérlettípus elnevezése
* Leírás – a bérlet jellemzői szövegesen meghatározva a következők alapján:
  + érvényesség napjainak száma
  + „napon belül felhasználható” szöveg
  + érvényesség alkalmainak száma, illetve a 999 érték esetében a „korlátlan” szöveg
  + „ alkalomra” szöveg
  + a bérlettípus ára kiegészítve a „Ft” pénznemmel.

Az utóbbi pedig egy legördülőmenüt állít össze, ahol a 0 értékhez az „Válasszon bérlettípust” szöveg kerül, a további bérlettípusok esetében az érték a bérlettípus ID-ja egy szóközzel elválasztva a bérlettípus elnevezésétől.

A „berletekLista” ismertetése:

Az oldalon elhelyezett „ervenyes” azonosítójú kapcsoló segítésével dönthet az adminisztrátor felhasználó, hogy valamennyi bérlet adatait vagy csak a jelenleg érvényes bérletek adatait kívánja megtekinteni.

Az érvényesség meghatározásához a ma0 és a ma24 változók kerültek bevezetésre, mert az eladott bérlet már a nap 0 órakor is érvényes és a lejárat napjának éjféléig is érvényes.

Az adatok szintén táblázatos formában jelennek meg:

* ID – bérlet azonosítója
* Ügyfél – ügyfél azonosítója és zárójelben az adatbázisban aktuálisan tárolt családneve szóközzel elválasztva a keresztnevétől
* Bérletnév – bérlettípus eladáskori elnevezése
* Eladnap – eladás ideje (dátum és idő)
* Kezdet – érvényesség kezdete (dátum)
* Napok – érvényesség napjainak száma
* Lejárat – lejáratának napja (dátum)
* Ár – a bérlet eladáskori ára
* Lehetőség - érvényesség alkalmainak száma, illetve a 999 érték esetében a „korlátlan” szöveg
* Érvényesség:
  + „felhasznált”, ha az alkalmak száma 0
  + „még nem érvényes”, ha az érvényesség első napja még nem volt
  + „már nem érvényes”, ha az érvényesség utolsó napja már elmúlt
  + „érvényes”, ha a mai nap az érvényesség első napja és utolsó napja között van

A „belepesekLista” ismertetése:

Az oldalon elhelyezett kapcsoló segítésével dönthet az adminisztrátor, hogy valamennyi belépésének vagy csak a bent lévőknek (kiléptetendő ügyfelek) belépésének adatait kívánja megtekinteni.

Az adatok mindkét esetben táblázatos formában jelennek meg:

* ID – belépés azonosítója
* Bérlet – bérlet azonosítója
* Ügyfél – az ügyfélnek az adatbázisban aktuálisan tárolt családneve szóközzel elválasztva a keresztnevétől
* Belépés – belépés ideje (dátum és idő)
* Kilépés – kilépés ideje (dátum és idő) vagy „kiléptet” gomb

A „beleptet” ismertetése:

Az eljárás paraméterként megkapja, melyik azonosítójú bérlettel történő belépést kell végrehajtani.

Az eljárás két részt foglal magában:

* a belépés tényének rögzítését (POST metódus)
* a hátralévő lehetőségek számának módosítását (PATCH metódus)

A „kileptet” ismertetése:

Az eljárás paraméterként megkapja, melyik azonosítójú belépés esetén kell a kiléptetést végrehajtani (PATCH metódus).

Az„ugyfelBerletekLista” ismertetése:

Az oldalon elhelyezett „ervenyes” azonosítójú kapcsoló segítésével dönthet az ügyfél, hogy valamennyi bérletének az adatait vagy a csak jelenleg érvényes bérletének adatait kívánja-e megtekinteni.

Az érvényesség meghatározásához a ma0 és a ma24 változók kerültek bevezetésre, mert az eladott bérlet már a nap 0 órakor is érvényes és a lejárat napjának éjfélekor is érvénybe van.

A lekérés a „berletek/” vagy „ervenyesberletek/” végpont ügyfélazonosítóval kiegészített címére küldött GET kérés ereményeként valósul meg.

Az adatok táblázatos formában jelennek meg:

* ID – bérlet azonosítója
* Ügyfél – ügyfél azonosítója
* Bérletnév – bérlettípus eladáskori elnevezése
* Eladás ideje – eladás ideje (dátum és idő)
* Kezdet – érvényesség kezdete (dátum)
* Napok – érvényesség napjainak száma
* Lejárat – lejáratának napja (dátum)
* Ár – a bérlet eladáskori ára
* Lehetőség - érvényesség alkalmainak száma, illetve a 999 érték esetében a „korlátlan” szöveg
* Beléptetés:
  + „felhasznált”, ha az alkalmak száma 0
  + „még nem érvényes”, ha az érvényesség első napja még nem volt
  + „már nem érvényes”, ha az érvényesség utolsó napja már elmúlt
  + „beléptet” gomb, ha a mai nap az érvényesség első napja és utolsó napja között van

Az érvényes bérletek adatai megjelennek egy legördülő menübe is, az ügyfél beléptetése innen is elvégezhető.

A „keres” ismertetése:

Akkor kerül lefutásra, ha a szűrőbe változás áll be. Mindössze az ugyfelekLista eljárást hívja meg.

Az „ugyfelAdatok” ismertetése:

A legördülőmenü segítségével ügyfelet kell választani (alapértelmezettként a 0-s azonosítójú – ügyfélrögzítésre alkalmas – érték van kijelölve).

Abban az esetben, ha ki lett valamelyik ügyfél választva, úgy adatai az űrlapba betöltésre kerülnek:

* ügyfél azonosítója (nem szerkeszthető mező)
* családneve
* keresztneve
* születési ideje
* neme
* e-mail címe
* telefonszáma
* címéhez tartozó irányítószám
* címéhez tartozó településnév
* címéhez tartozó lakcím adatok (közterület neve, jellege, házszám, stb. egy mezőben)
* hírlevelet kér-e
* jelszó megváltoztatására lehetőséget adó jelszómező (a jelszó nem kerül tárolásra, csak az abból képzett kód)

Ha a „0” azonosító van kiválasztva, üres űrlap jelenik meg.

Bérleteladás ismertetése

Az ügyfél kiválasztása után az adminisztrátornak ki kell választani a kért bérlettípust és beállítani az érvényesség első napját.

A legördülő menüből kiválasztott bérlettípus azonosítójával lekérésre kerülnek a bérlettípus általános adatai:

* érvényesség napjainak száma
* bérlet ára
* bérlettel igénybe vehető alkalmak száma
* bérlet neve.

A fenti adatok bírtokában a bérletek táblába rögzítésre kerül:

* az ügyfél azonosító száma
* bérlet neve
* érvényesség kezdete (amelyet az adminisztrátor állított be)
* érvényesség napjainak száma
* bérlet ára
* bérlettel igénybe vehető alkalmak száma

A bérlet adatai azért kerülnek ebben a táblában rögzítésre, mert így el lehet kerülni, hogy a bérlettípus szerkesztése visszamenőleges változtatást okozzon.

A sikeres bérleteladás a „Az ügyfélnek a bérlet eladásra került” üzenettel nyugtázásra kerül, majd újra betöltődik az ügyfél bérleteinek listája.

Rögzítés gomb ismertetése

A rögzítés gomb segítségével történik az új ügyfél rögzítése (azonosító 0, az ügyfél valós azonosítója a rögzítés során az adatbázisból kerül meghatározásra) vagy a 0-tól eltérő azonosítójú ügyfél adatainak módosítása.

A rögzítés folyamatának első lépéseként bizonyos mezők ellenőrzésre kerülnek:

* családnév mező akkor fogadható el, ha legalább 2 karaktert tartalmaz
* keresztnév mező akkor fogadható el, ha legalább 2 karaktert tartalmaz
* születési idő akkor fogadható el, ha nem üres (korhatár nincs meghatározva)
* e-mail cím akkor fogadható el, ha legalább 6 karaktert tartalmaz (jelen változatban nem kerül ellenőrzésre, hogy @ jel kötelezően csak egy lehet benne, a . után a végén legalább két karakter szerepeljen, stb.)
* telefonszám akkor fogadható el, ha legalább 10, de legfeljebb 12 karaktert tartalmaz
* irányítószám akkor fogadható el, ha értéke 1001-9999 között van
* település neve akkor fogadható el, ha legalább 2 karaktert tartalmaz
* lakcím akkor fogadható el, ha legalább 1 karaktert (akár egy – karakter) tartalmaz
* ha jelszó megadásra kerül, akkor csak akkor fogadható el, ha legalább 5 karaktert tartalmaz; új ügyfél esetén kötelező megadni

Ha a fenti ellenőrzés rendben megtörtént, POST metódus segítségével az új ügyfél adatai rögzítésre, PATCH metódus segítségével pedig módosításra kerülnek. A módosítás esetében is két lehetőség van: az egyik, amikor a régi jelszó továbbra is változatlan marad, a másik esetben a jelszó –akár csak önmagában – módosul. Ez utóbbi lehetőséget ad elfelejtett jelszó esetén új jelszó beállítására.

A rögzítés sikerességét „Az új ügyfél adatai rögzítésre kerültek!” vagy „Az ügyfél adatai módosításra kerültek!” üzenet jelzi vissza.

Beléptet gomb ismertetése

A beléptet gomb megnyomására a kiválasztott bérlet azonosítójának megfelelő bérlet azonosítójával – mint paraméterrel – meghívásra kerül a beleptet eljárás.

Üres szűrő gomb ismertetése

A gomb megnyomására a szűrő tartalma törlődik, üressé változik és újra lekérésre kerül (az összes ügyfelet tartalmazó) ügyféllista.

# A bérletszerkesztő (tulajdonosi) felület leírása

Megjegyzés: bérlet szó alatt értjük a terem igénybevételére jogosító – akár csak egy alkalomra, speciális óratípusra szóló – napijegyeket és a több alkalomra is érvényes belépőket.

A „staff-berlet.html” elnevezésű oldal stílusát a „staffstilus.css” fájl tartalmazza, a szükséges scriptek a „staff-berlet.js” fájlban találhatók.

Az oldal tetején az edzőterem logója található, ezt követi a jelenleg értékesíthető bérlettípusok tábla; ez az ID, Elnevezés, Leírás, Ár oszlopokból áll.

A következő blokk a „Bérlettípus hozzáadása/módosítása/törlése” rész.

Egy legördülő menü segítségével kiválasztható bármely, a fenti táblázatban található, már rögzített bérletfajta, ha azok módosítása vagy törlése válik szükségessé. Alaphelyzetben az „Új bérlettípus rögzítése” van kijelölve, így a meglévők mellé bizonyos korlátok mellett tetszőleges tartalmú belépésre jogosító bérlet rögzíthető.

A rögzítés során az alábbi kötöttségeket kell figyelembe venni:

* „Bérletnév” mező nem lehet üres
* „Érvényesség napjai” értéke 1 és 366 közé eshet, így a leghosszabb érvénytartam egy év lehet
* „Maximális belépések száma” értéke 1 - 999 közé eshet, a 999 számmal a korlátlan alkalommal történő belépésre jogosító bérlet kerül megjelölésre
* „Ár” mező értéke 1 Ft és 999 999 Ft közé eshet

A „Rögzítés” gomb menti az adatokat (függetlenül attól, hogy csak módosítás volt vagy új bérlet rögzítése), a „Törlés” gomb – kiválasztott bérlettípus esetén, megerősítés után – törli az adatbázisból, de a mezőkben meghagyja az értékeket, ha esetleg mégis újra kívánják rögzíteni (ebben az esetben viszont új ID-t fog kapni).

A következő blokk a „Bérletek listája”, amely alapesetben a jelenleg érvényes bérleteket listázza ki. Az alapesteben bejelölt „Csak érvényes bérletek” kapcsoló kikapcsolásával valamennyi eladott bérlet adatai láthatóvá válnak.

Az oldal működéséért a „staff-berlet.js” fájl felel.

Az alapértelmezett hoszt a localhost 4000-es portja. (A portot más feladat során nem használtuk, ezért ez lett kijelölve.)

Induláskor lefut a **berlettipusokLista** és a **berletekLista** funkciók.

A „berlettipusokLista” ismertetése:

Feladata két lista (berlettipusokLista és berlettipusokLista1) elkészítése, melyből az előző eredménye egy táblázatos formátumú megjelenítés és adatbázisban tárolt adatok felhasználásával az alábbi oszlopok szerint:

* ID – bérlettípus azonosítója
* Elnevezés – bérlettípus elnevezése
* Leírás – a bérlet jellemzői szövegesen meghatározva a következők alapján:
  + érvényesség napjainak száma
  + „napon belül felhasználható” szöveg
  + érvényesség alkalmainak száma, illetve a 999 érték esetében a „korlátlan” szöveg
  + „ alkalomra” szöveg
  + a bérlettípus ára kiegészítve a „Ft” pénznemmel.

Az utóbbi pedig egy legördülőmenüt állít össze, ahol a 0 értékhez az „Új bérlettípus rögzítése”, a további bérlettípusok esetében az érték a bérlettípus ID-ja egy szóközzel elválasztva a bérlettípus elnevezésétől.

A „berletekLista” ismertetése:

Az oldalon elhelyezett „ervenyes” azonosítójú kapcsoló segítésével dönthet a (tulajdonos) felhasználó, hogy valamennyi bérlet adatait vagy csak a jelenleg érvényes bérlet adatait kívánja-e megtekinteni.

Az érvényesség meghatározásához a ma0 és a ma24 változók kerültek bevezetésre, mert az eladott bérlet már a nap 0 órakor is érvényes és a lejárat napjának éjféléig is érvényes.

Az adatok szintén táblázatos formában jelennek meg:

* ID – bérlet azonosítója
* Ügyfél – ügyfél azonosítója és zárójelben az adatbázisban aktuálisan tárolt családneve szóközzel elválasztva a keresztnevétől
* Bérletnév – bérlettípus eladáskori elnevezése
* Eladnap – eladás ideje (dátum és idő)
* Kezdet – érvényesség kezdete (dátum)
* Napok – érvényesség napjainak száma
* Lejárat – lejáratának napja (dátum)
* Ár – a bérlet eladáskori ára
* Lehetőség - érvényesség alkalmainak száma, illetve a 999 érték esetében a „korlátlan” szöveg
* Érvényesség:
  + „felhasznált”, ha az alkalmak száma 0
  + „még nem érvényes”, ha az érvényesség első napja még nem volt
  + „már nem érvényes”, ha az érvényesség utolsó napja már elmúlt
  + „érvényes”, ha a mai nap az érvényesség első napja és utolsó napja között van

A berletAdatok ismertetése:

Az eljárás az űrlapmezőbe betölti a kiválasztott bérlet adatait, vagy új felvétele esetén üres űrlappal indít.

A torol ismertetése:

Az eljárás egy kiválasztott bérlet esetében – egy megerősítő üzenet után – az adatbázisból törli a bérlet adatait. A törölt bérlet adatai az űrlapon betöltve megmaradnak, hogy téves törlés esetén újra lehessen rögzíteni, de ebben az esetben már új ID-t fog kapni.

Az urlaptorles ismertetése:

Az eljárás az űrlap mezőinek adatait üresre állítja.

Rögzítés gomb megnyomása:

A rögzítés gomb az új bérlettípus adatainak rögzítésére vagy – kiválasztott bérlettípus esetén – a meglévő bérlettípus adatainak módosítások elmentésére szolgál.

A rögzítés gomb megnyomása esetén az alábbi ellenőrzések kerülnek elvégzésre:

* bérletnév legalább egy karakter hosszú-e
* érvényesség napjainak száma 1 és 366 között van-e
* alkalmak száma 1 és 999 között van-e
* az ár legalább 1 Ft

Ha a fenti feltételek teljesülnek, a bérlettípus rögzítése (módosítása) megtörténik, valamint az űrlapmezők üres állapotúak lesznek.

# Az ügyfél által megtekinthető felület leírása

Az „index.html” elnevezésű oldal stílusát a „stilusok.css” fájl tartalmazza, a szükséges scriptek az „index.js” fájlban találhatók.

Az oldalon bejelentkezés nélkül bárki megtekinthető a bérlettípusok táblázatát, amelyet a berlettipusokLista függvény jelenít meg.

A „berlettipusokLista” ismertetése:

Feladata egy táblázatos formátumú megjelenítés és adatbázisban tárolt adatok felhasználásával az alábbi oszlopok szerint:

* ID – bérlettípus azonosítója
* Elnevezés – bérlettípus elnevezése
* Leírás – a bérlet jellemzői szövegesen meghatározva a következők alapján:
  + érvényesség napjainak száma
  + „napon belül felhasználható” szöveg
  + érvényesség alkalmainak száma, illetve a 999 érték esetében a „korlátlan” szöveg
  + „ alkalomra” szöveg
  + a bérlettípus ára kiegészítve a „Ft” pénznemmel.

Az oldalon egy egyszerű űrlap található, melyen két beviteli mező van: az ügyfél azonosítója és a hozzá tartozó jelszó.

Az ügyfél az adatai megtekintéséhez megadja az azonosítóját és a jelszavát. A két adat helyességét az app.js fájl az adatbázis segítésével végzi el, a hibaüzenetek egy részét is innen fogja megkapni az ügyfél. Az index.js fájl csak a mezők üres állapotát vizsgálja, mely során az „Üres ügyfélazonosító vagy jelszó!” hibaüzenet érkezhet.

Bejelentkezés (kijelentkezés) gomb megnyomása:

Az eljárás a „bejelentkezve” változó állapotát vizsgálja. Annak igaz értékekor egy megerősítendő üzenetre – „Biztosan ki szeretne jelentkezni?” – adott beleegyező válasz esetén:

* törli a sessionStorage állapotát (tokent és az ügyfél azonosítóját),
* üresre állítja a két beviteli mezőt,
* „Belépés” feliratot állít vissza a gombra,
* feloldja az ügyfélazonosító mező szerkeszthetőségét,
* „Sikeres kijelentkezés” üzenetet ad fel,
* a lekerdezes eljárás alapján a megjelenített adatokat törli,

majd visszatér a meghívás helyéhez.

A következő részben a két mező üres állapotát vizsgálja; ha bármelyik mező (azonosító, jelszó) üres, akkor hibaüzenetet követően tér vissza a meghívás helyéhez.

Ha a fenti részekben nem került az eljárás megszakításra a visszatéréssel, megtörténik a bejelentkezés feltételeinek (jó azonosító – jelszó páros) ellenőrzése. Amennyiben a bejelentkezés sikeres, válaszban megérkezik a token.

A token megléte esetén

* sessionStorage-ben tárolja a tokent,
* sessionStorage-ben tárolja az ügyfél azonosítóját,
* letiltja az ügyfélazonosító mező szerkeszthetőségét,
* jelszó mezőt üresre állítja,
* bejelentkezve változó értékét igazzá teszi,
* „Kilépés” lesz a gomb szövege,
* lefut a lekerdezes eljárás.

A „lekerdezes” ismertetése:

Az eljárás három további eljárást hív meg:

* ugyfelAdatok,
* ugyfelBerletekLista,
* belepesekLista.

Az „ugyfelAdatok” ismertetése:

Az eljárás ellenőrzi, hogy van-e a sessionStorage-ban ügyfélazonosító és token tárolva, ennek hiánya esetén a „Kérem, jelentkezzen be!” üzenet kerül ki az oldalra és visszatér a meghívás helyére az eljárás.

A token birtokában történik meg az adatok lekérése, majd szövegesen a megjelenítése:

* Ügyfél azonosítóm: <ügyfélazonosító>
* Nevem: <családnév és keresztnév>
* Születési időm: <születési idő>
* Nemem: <férfi vagy nő>
* E-mail címem: <e-mail cím>
* Címem: <irányítószám, település, lakcím>
* Telefonom: <telefonszám>
* Hírlevelet kérek / Hírlevelet nem kérek.

Az „ugyfelBerletekLista” ismertetése:

Az oldalon elhelyezett „ervenyes” azonosítójú kapcsoló segítésével dönthet az ügyfél, hogy valamennyi bérletének az adatait vagy a csak jelenleg érvényes bérletének adatait kívánja-e megtekinteni.

Az érvényesség meghatározásához a ma0 és a ma24 változók kerültek bevezetésre, mert az eladott bérlet már a nap 0 órakor is érvényes és a lejárat napjának éjfélekor is érvénybe van.

A lekérés csak a token birtokában valósul meg az „azonositottberletek/” vagy „azonositottervenyesberletek/” végpont ügyfélazonosítóval kiegészített címére küldött GET kérés ereményeként.

Az adatok táblázatos formában jelennek meg:

* ID – bérlet azonosítója
* Bérletnév – bérlettípus eladáskori elnevezése
* Vásárlás ideje – eladás ideje (dátum és idő)
* Kezdet – érvényesség kezdete (dátum)
* Napok – érvényesség napjainak száma
* Lejárat – lejáratának napja (dátum)
* Ár – a bérlet eladáskori ára
* Lehetőség - érvényesség alkalmainak száma, illetve a 999 érték esetében a „korlátlan” szöveg
* Állapot:
  + „felhasznált”, ha az alkalmak száma 0
  + „még nem érvényes”, ha az érvényesség első napja még nem volt
  + „már nem érvényes”, ha az érvényesség utolsó napja már elmúlt
  + „érvényes”, ha a mai nap az érvényesség első napja és utolsó napja között van

A „belepesekLista” ismertetése:

Az oldalon elhelyezett kapcsoló segítésével dönthet az ügyfél, hogy valamennyi belépésének vagy csak az utolsó 5 belépésének adatait kívánja-e megtekinteni.

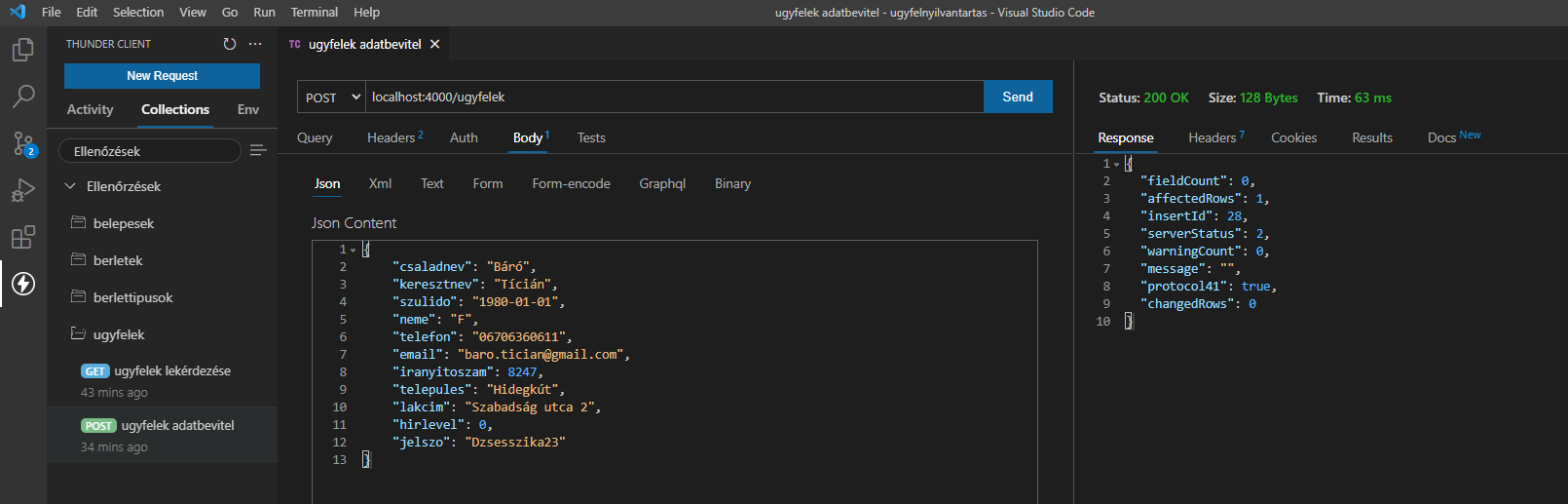
A lekérés csak a token birtokában valósul meg az „azonositottugyfelbelepesek/” vagy „azonositottugyfelbelepesek5/” végpont ügyfélazonosítóval kiegészített címére küldött GET kérés ereményeként.

Az adatok mindkét esetben táblázatos formában jelennek meg:

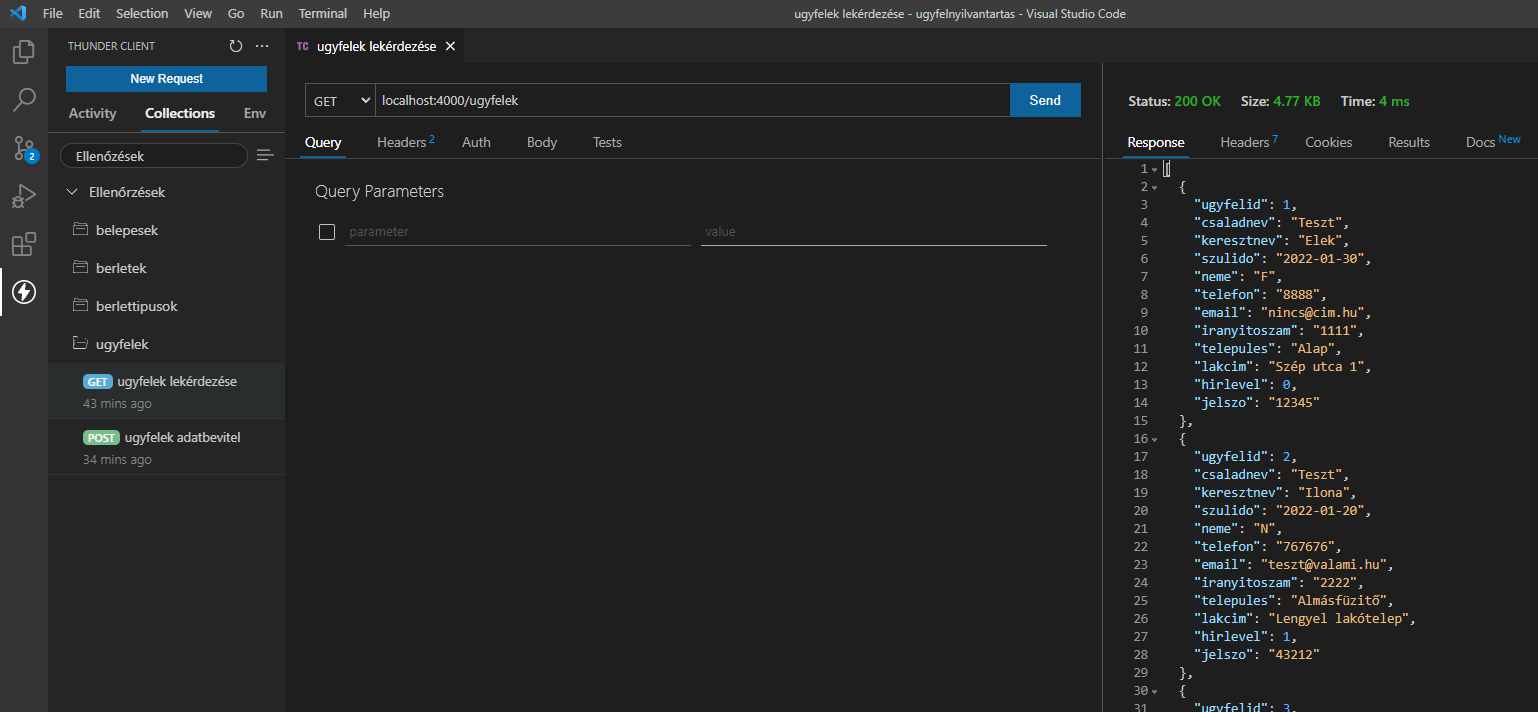
* ID – belépés azonosítója
* Bérlet – bérlet azonosítója
* Belépés – belépés ideje (dátum és idő)
* Kilépés – kilépés ideje (dátum és idő) vagy „még bent vagyok” szöveg

# Az adatok küldésének és lekérésének tesztelése

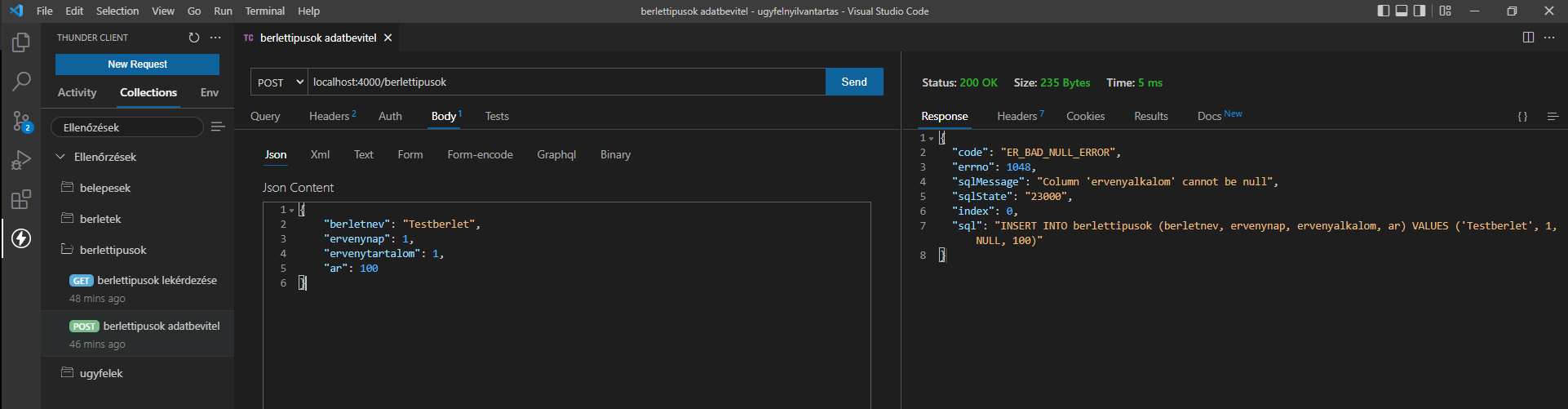
**Ügyfelek adatbevitele**



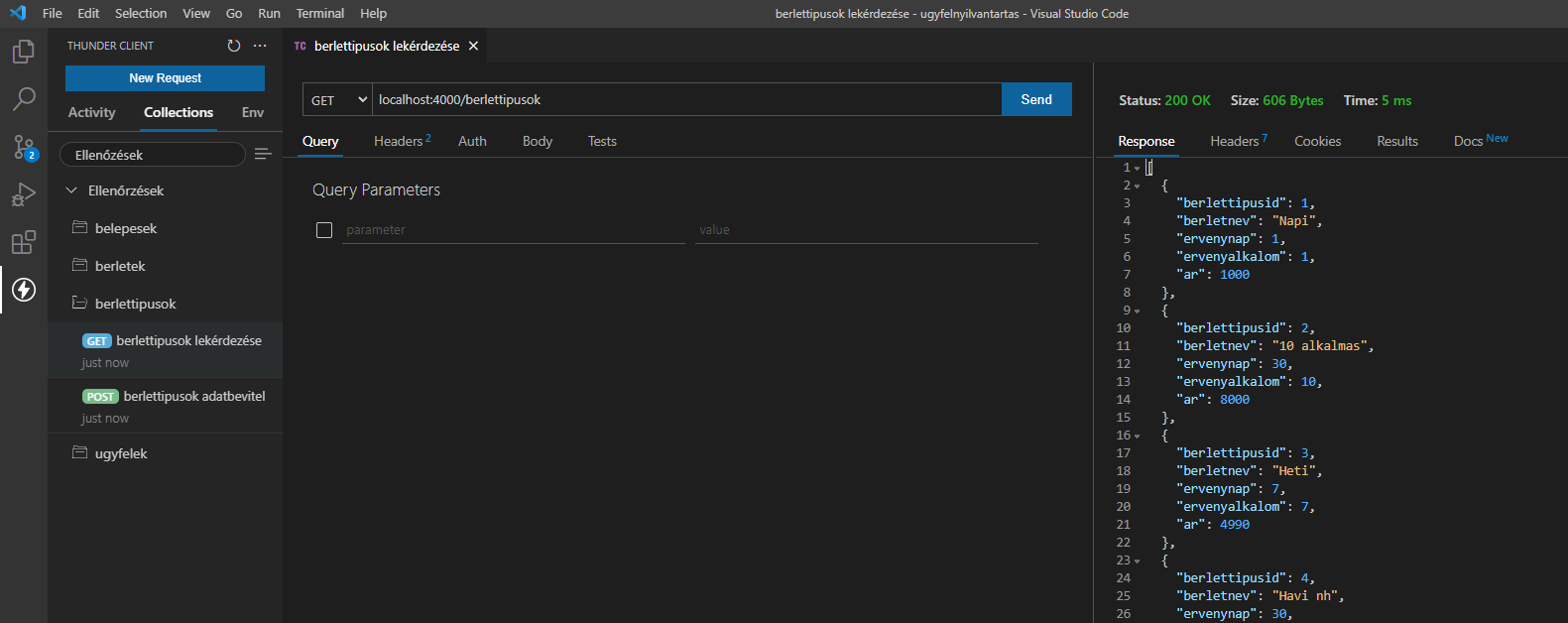
**Ügyfelek lekérdezése**



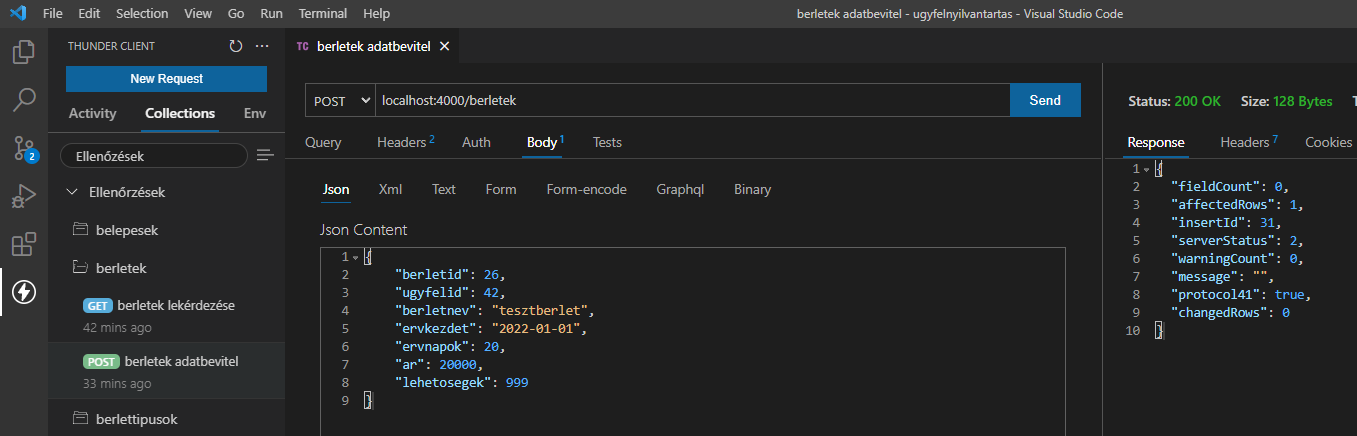
**Bérlettípusok adatbevitele**



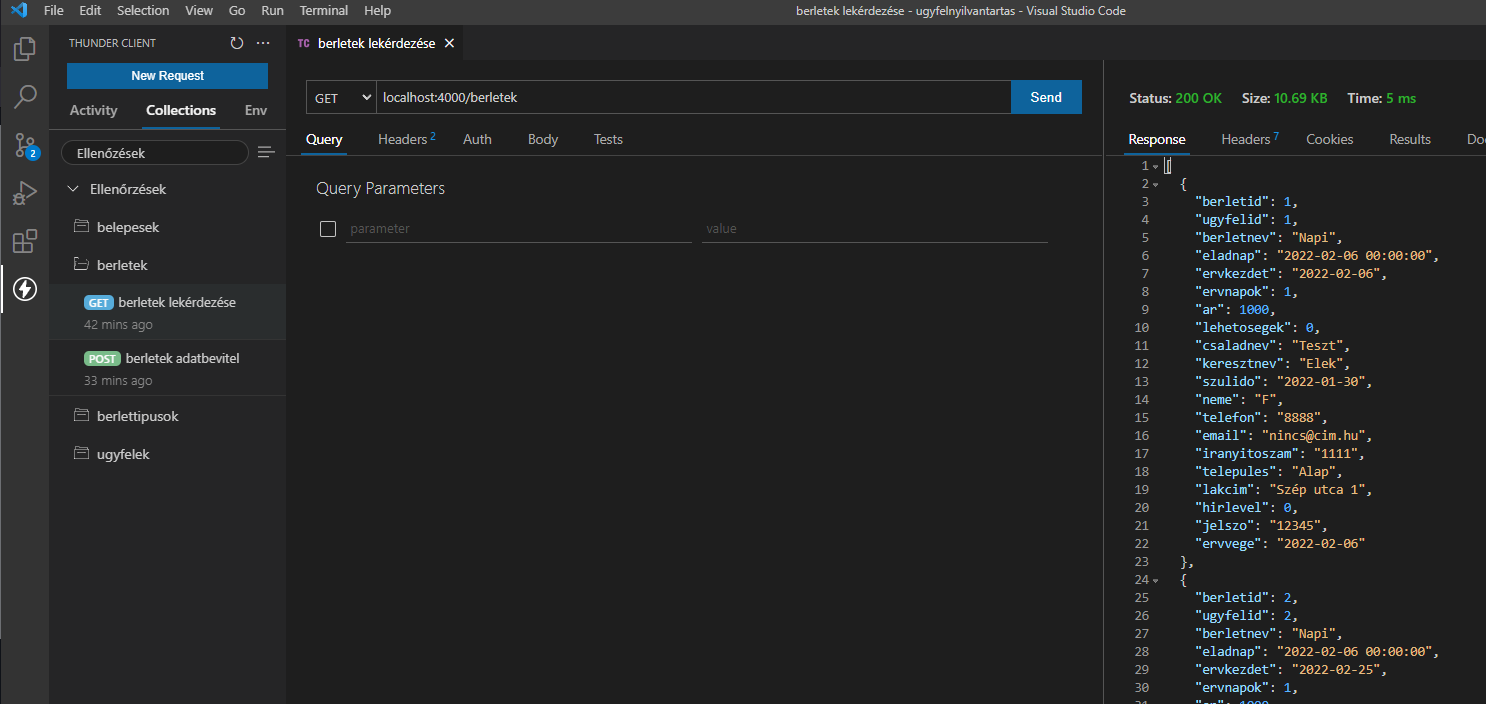
**Bérlettípusok lekérdezése**



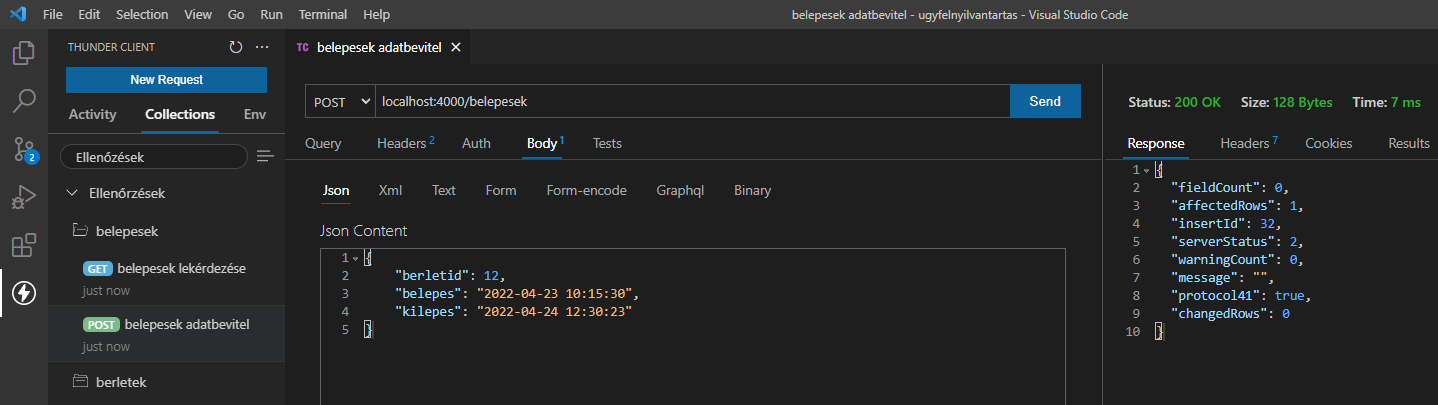
**Bérletek adatbevitel**



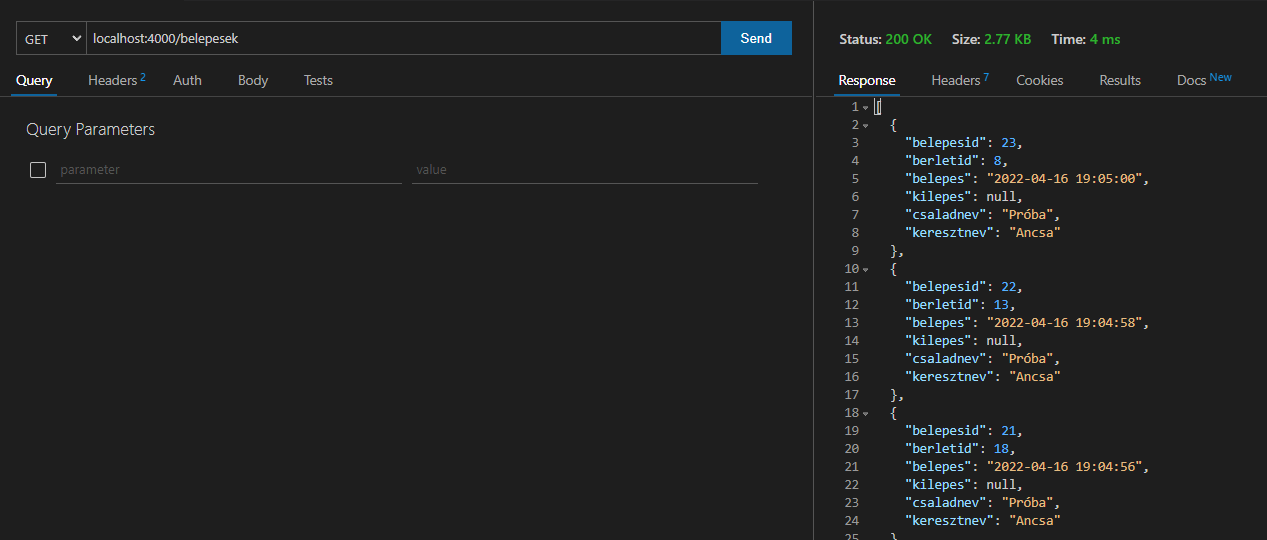
**Bérletek lekérdezése**



**Belépések adatbevitele**



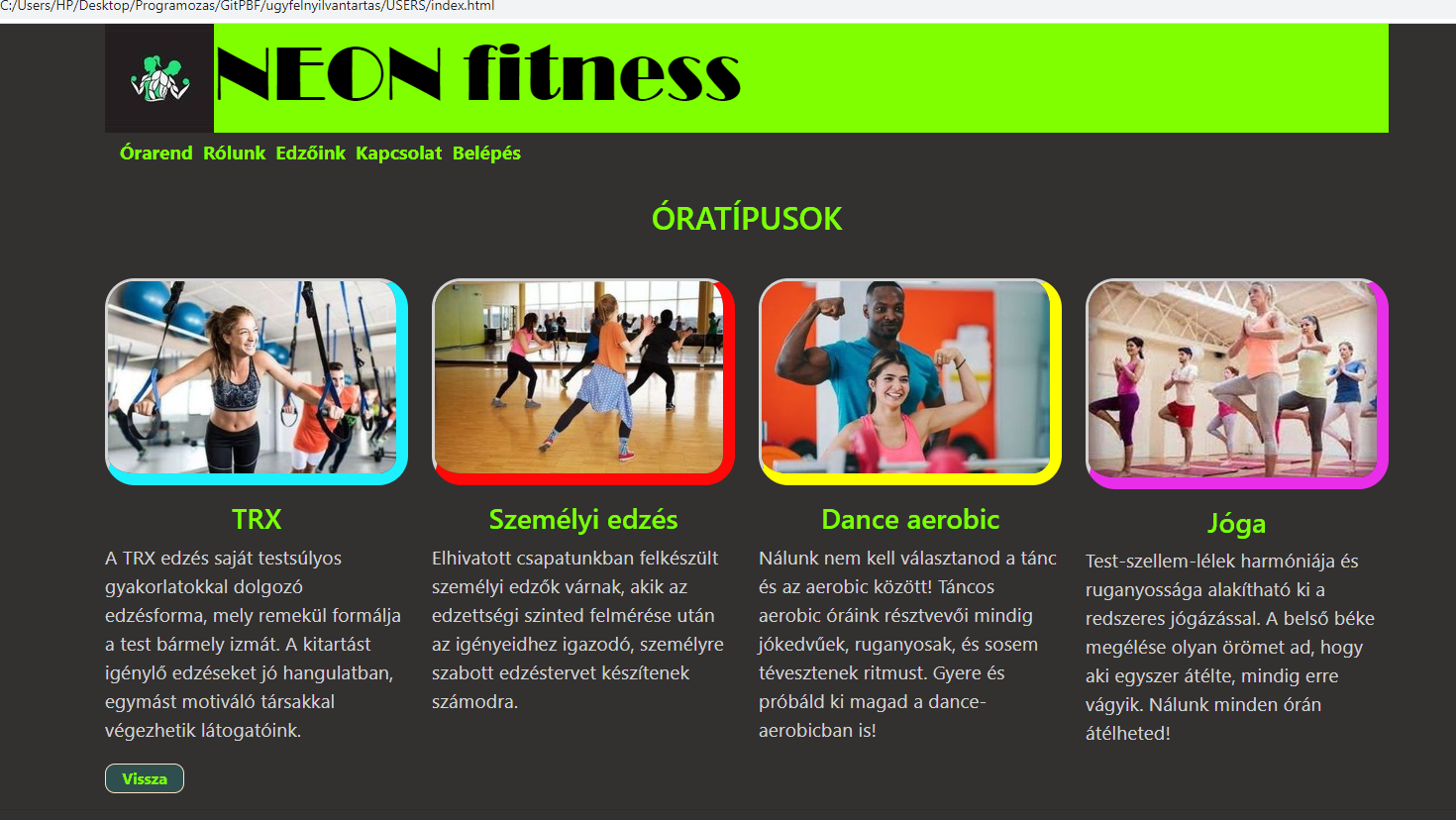
**Belépések lekérdezése**



# Az edzőterem kliens oldalának elkészítése

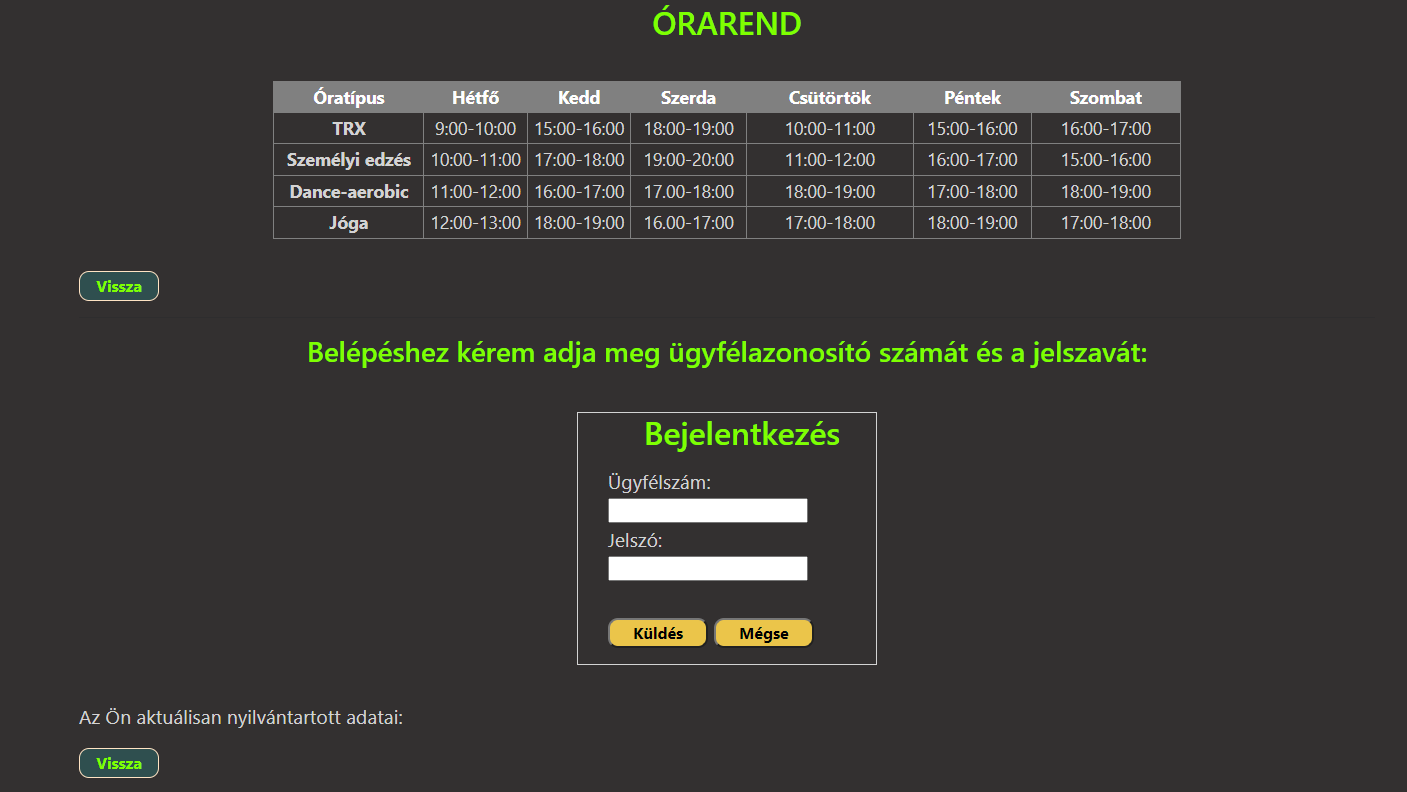
Bootstrap alkalmazással reszponzív oldalt készítettünk, melynek tervezésénél kiemelt szempont volt az egyszerűség, átláthatóság és a gyors információ elérés biztosítása az ügyfeleknek. Mindezt a weboldal egyetlen lapjának felső sávjában megjelenő fülek, valamint a hozzájuk rendelt tartalmak végén elhelyezett visszalépő gombokkal valósítottuk meg.

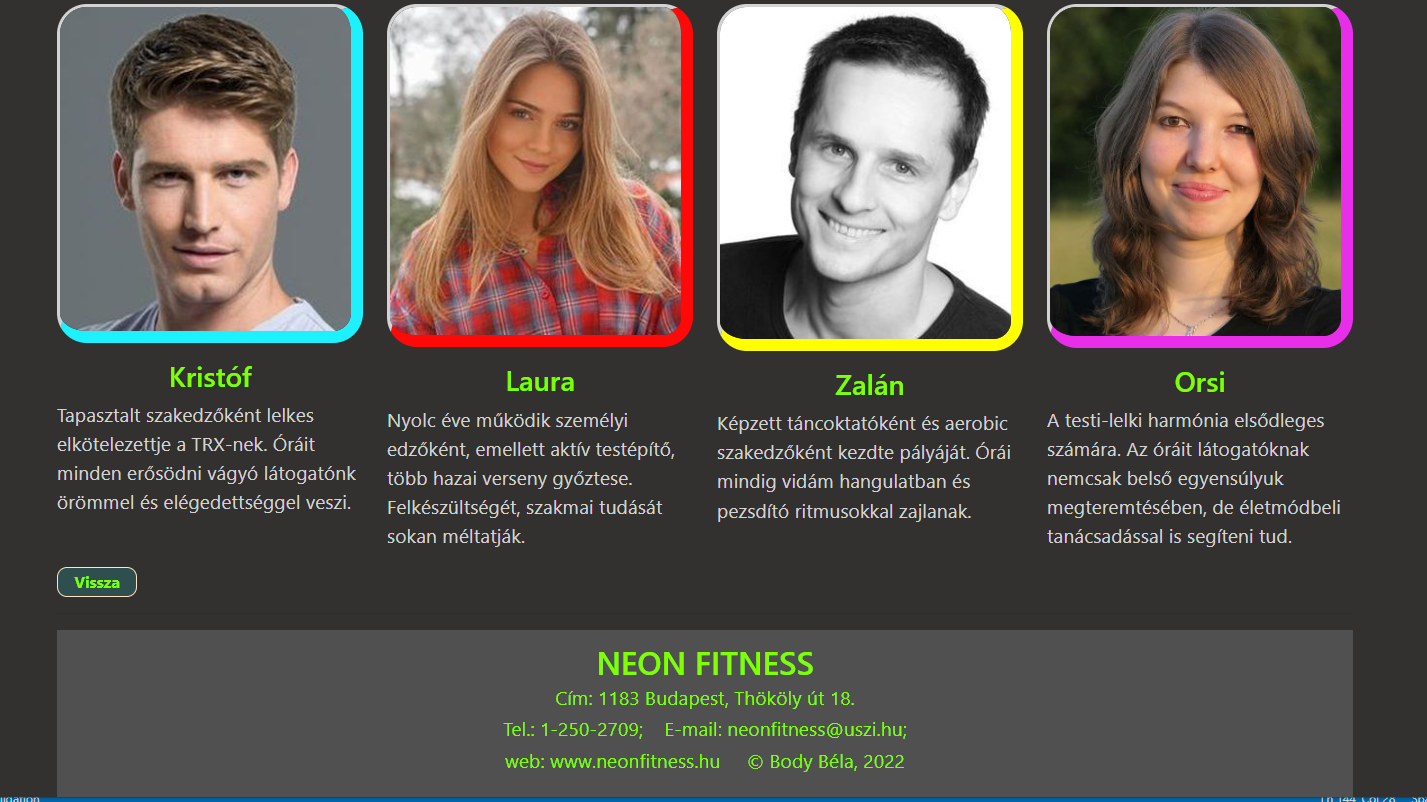
Az oldal megjelenése széles monitoron:



A kliens oldalon az ügyfelek ellenőrizhetik az edzőterem által nyilvántartott személyes adataik pontosságát, valamint aktuális bérletük érvényességi idejét, s a még felhasználható alkalmak számát. Ehhez az oldalon elérhető Bejelentkezés formot kell első lépésben kitölteniük az ügyfélszámuk és a jelszavuk megadásával.

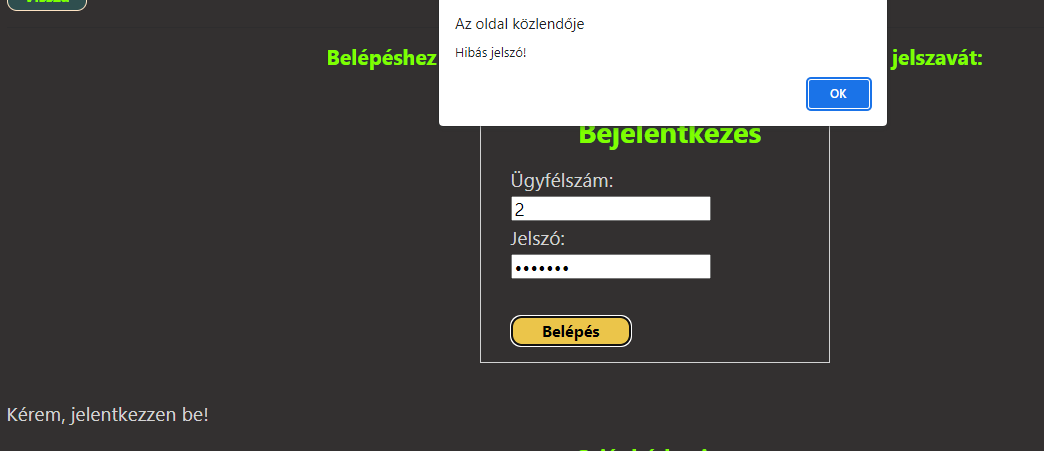
A küldés gombra kattintva az API ellenőrzi az adatok pontosságát, majd visszaküldi az adott ügyfél aktuálisan nyilvántartott adatait.



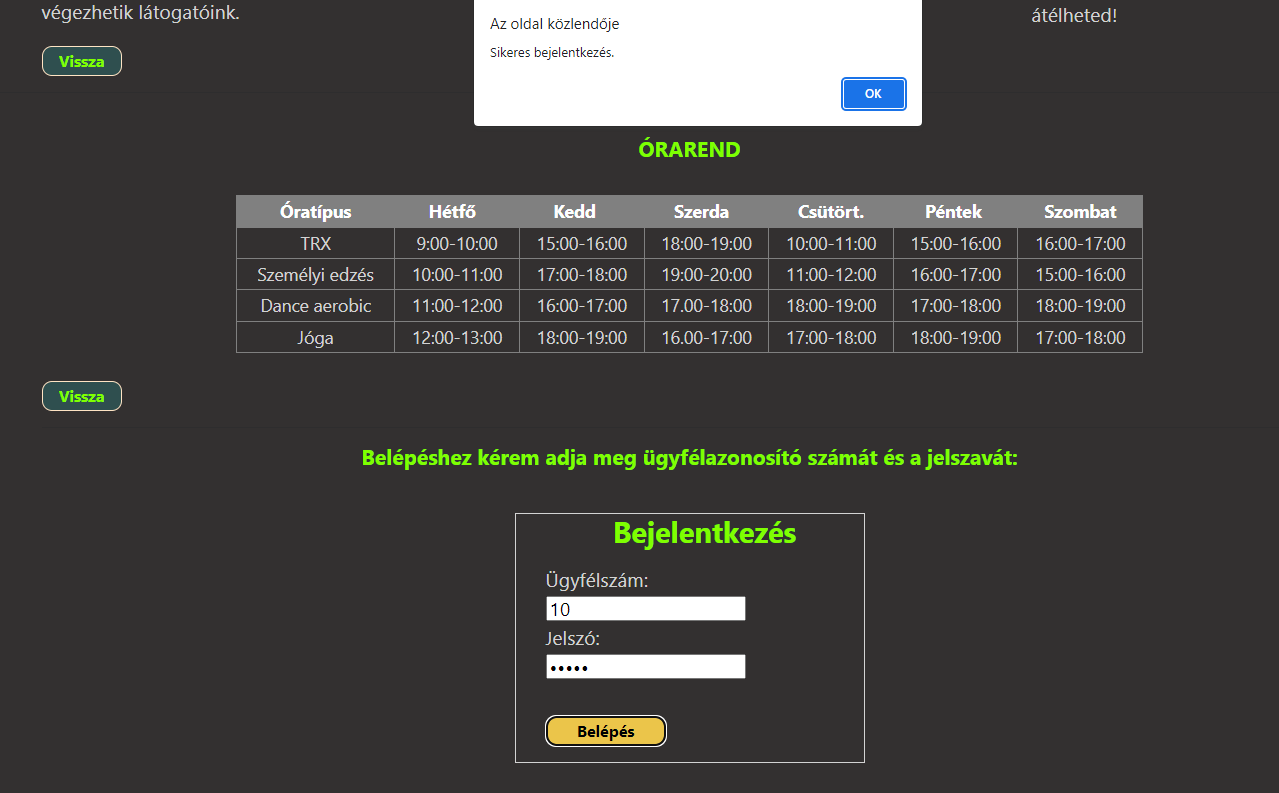


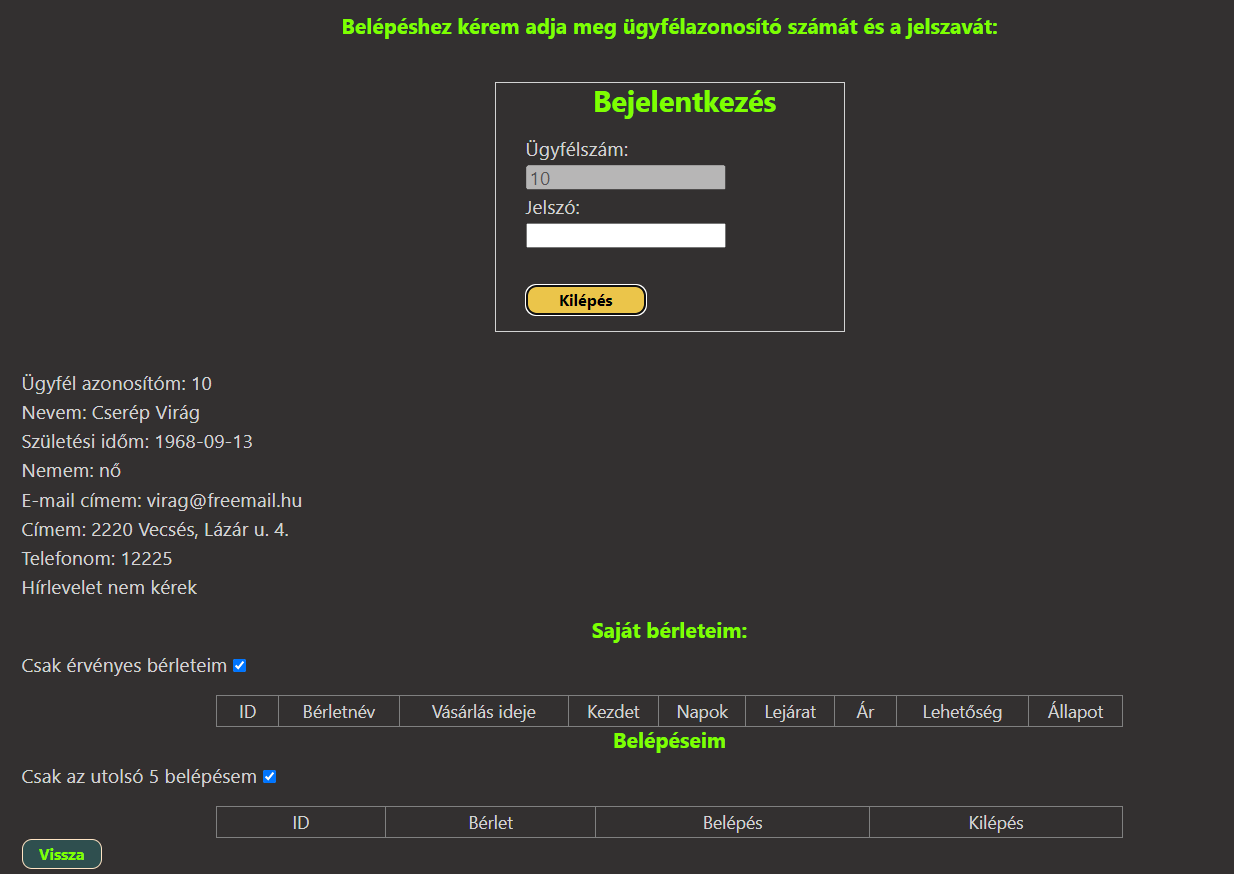
Az oldal adatlekérési funkciójának ellenőrzése:

Hibás jelszó esetén:

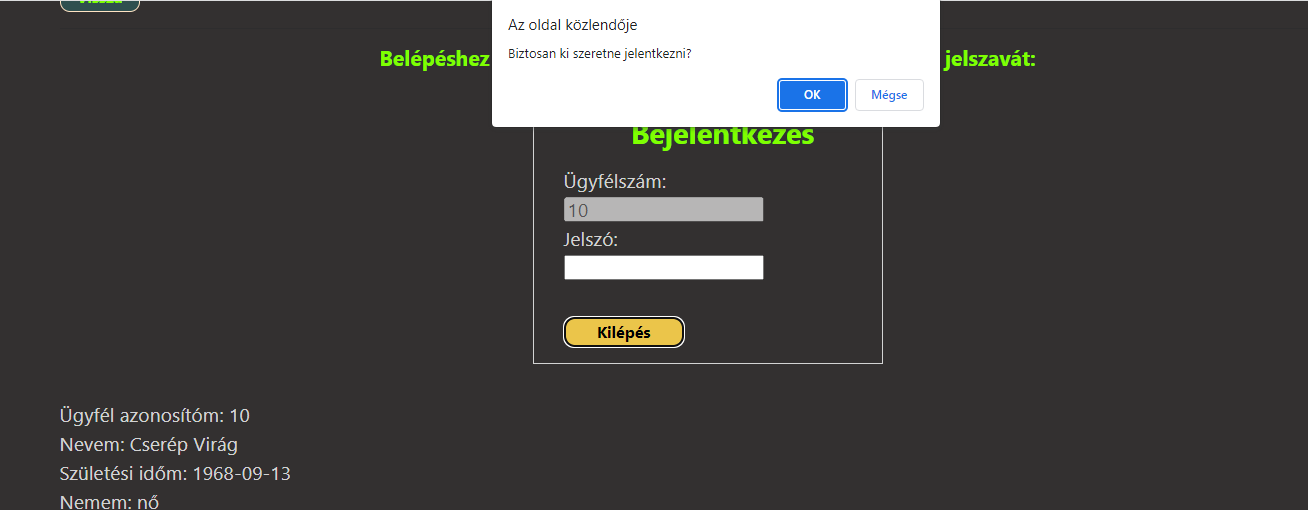


Helyes jelszó esetén:





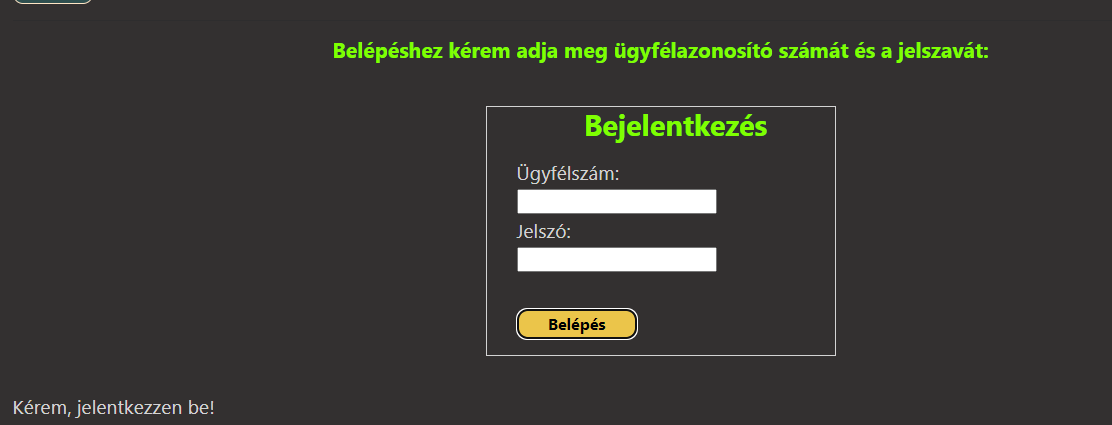
Kilépés a kijelentkezés megerősítésével:



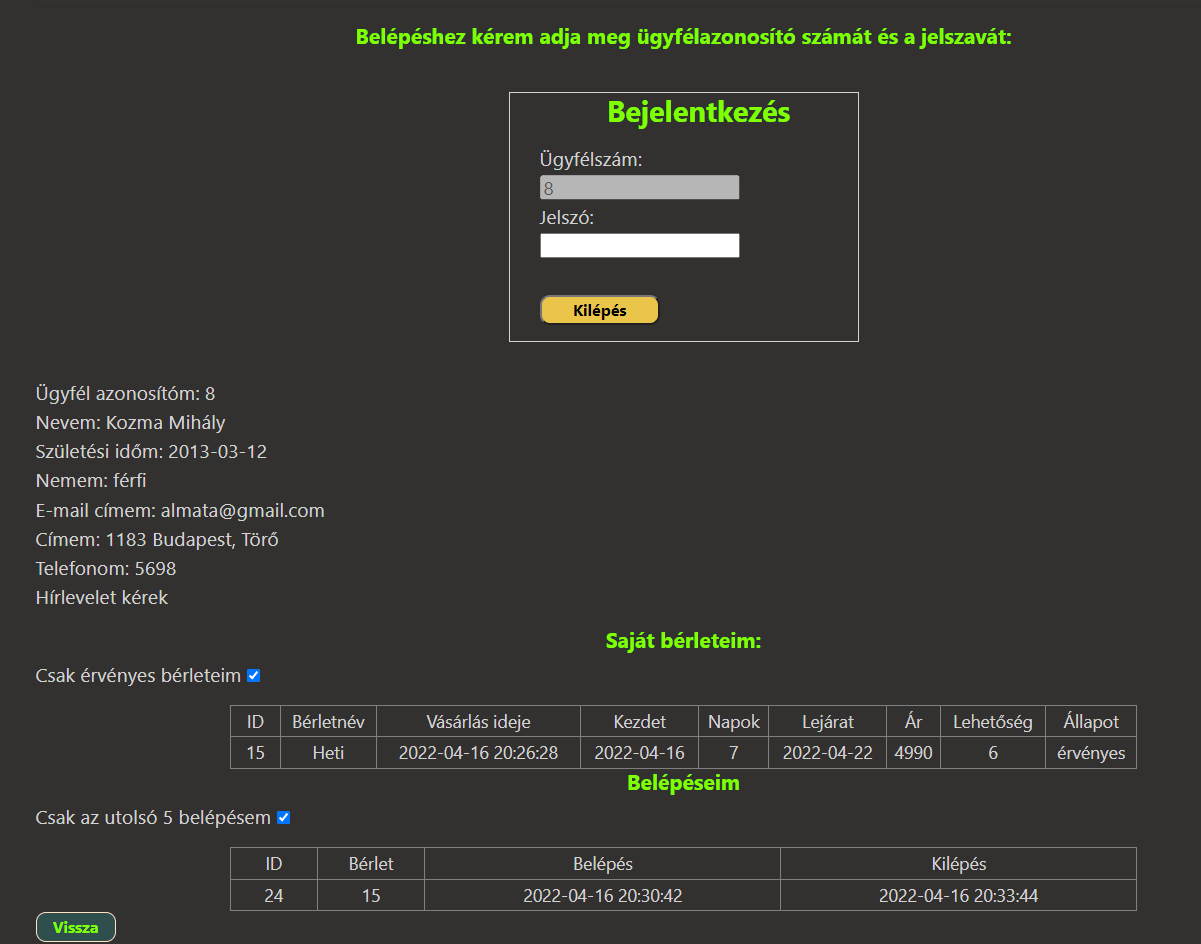
A kijelentkezés visszaigazolása:



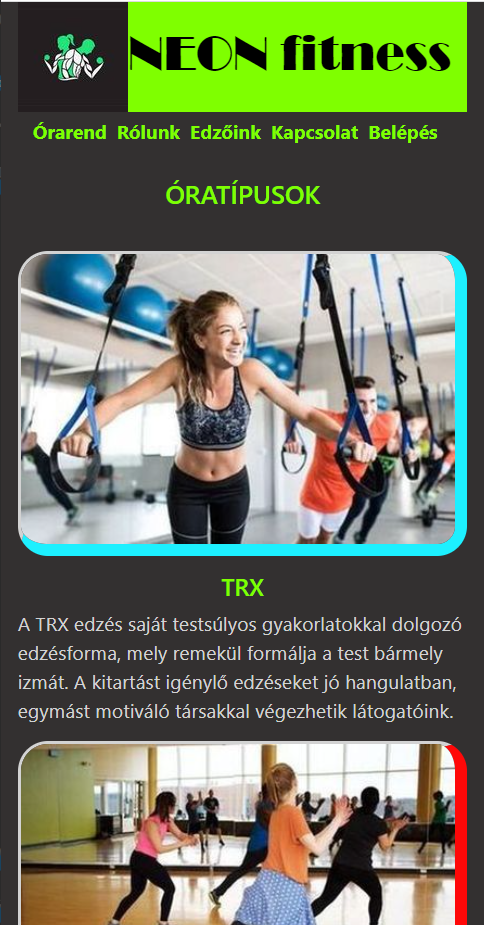
A bejelentkezési form üresre állítása:



Másik ügyfél bejelentkezése:



A kliens oldal eleje és a bejelentkezési form mobil eszközön:

# Zárógondolatok a fenti ügyfélnyilvántartó program elkészítésének folyamatáról

A feladat megvalósítására csoportot alakítottunk, ami esetünkben a kiírásnak megfelelően három főből állt. Már az elején megállapodtunk abban, hogy minden csoporttag külön, meghatározott feladatrészt készít el a programozás során, elsősorban olyat, amit szívesen végez. Így gyorsan gazdára találtak a főbb megvalósítási elemek az alábbiak szerint:

|  |  |
| --- | --- |
| **Megoldásra váró feladatok** | **Felelős** |
| kezdeti tervezés, ötletelés,  adatbázis tervezése, tartalma | közös munka |
| színpaletta választás,  logótervezés | közös munka |
| Frontend  (HTML) kliens oldal összeállítása, képek válogatása  User oldal JavaScript programozása | Hakkel H. – Bagi F.  Petrich B. F. |
| Backend  API,  Staff-admin oldalak programozása  Staff-admin HTML oldalak | Petrich B. F.  Hakkel H. |
| Ellenőrzések,  tesztelés | Bagi F.  Hakkel H. |
| Dokumentáció elkészítése | közös munka |

Az önálló feladatok előtt közösen összegyűjtöttük az adatbázishoz szükséges adattípusokat, majd egymáshoz kapcsolódó adattáblákba rendeztük azokat, egy adatbázis kezelő program segítségével.

A rögzített adatokat tesztelnünk is kellett mind a felhasználó, mind a tulajdonos igényeinek szempontjából. Mikor ez jól működött, rátértünk a következő feladatrészre: az egyéni feladatokra.

Mivel ezek többnyire önálló munkát igényeltek, az elkészült részfeladatok egymás közötti megosztásához a GitHub felületét, míg folyamatok közben történő gyors egyeztetésekhez az internetes levelező rendszereket, valamint a képernyő-megosztó felületeket használtuk.

A nehézségekről:

A közös munka során talán a legnagyobb nehézséget a GitHub gördülékeny használata jelentette. Ebben az eltérő otthoni eszközfelszereltségünk is szerepet játszhatott; a felülethez való csatlakozás nem ment mindig egyszerűen, s a létrehozott szerverkapcsolat is sokszor cserbenhagyott minket. Ami az egyik csoporttagunknál jól működött, az a másiknál csak részlegesen. Így kissé nehézkes volt az előrehaladás, s olykor elbeszéltünk egymás mellett.

Talán a csoport vegyes életkori összetétele okán is időbe telt, mire annyira összerázódtunk, hogy pontosan tudtuk követni egymás feladatait, vagy mondandóját a távkapcsolat során is.

E mellett egymás elérése, a közös megbeszélések időpontjának egyeztetése is komoly logisztikát igényelt, mivel a csoporttagok szabad idejének csak töredék része mutatott átfedést.

Az egyéni és közös munka során számos új és újabb ötlet merült fel a csoporttagok részéről az addigi tartalmak kiegészítését, logikusabb, vagy esztétikusabb megjelenítését illetően, ezért olykor párhuzamosan is dolgoztunk egy-egy részfeladaton, keresve a legjobb megoldást. Az eközben megélt esetleges kudarcokon is túl kellett tudni lépnünk.

A pozitívumokról:

Egyértelmű pozitívumként éltük meg, hogy egy-egy elakadásnál volt kihez fordulni, s kérdezhettünk egymástól.

Mindannyian úgy ítéltük meg, hogy a legnagyobb léptékkel akkor tudtunk haladni a munkában, mikor sikerült összeülni és összefésülni a változtatásokat, valamint személyesen is egyeztetni a további lépéseket.

Az elkészült feladatrészek szaporodása, s az oldalak testet öltő látványa mindig újabb energiát és türelmet adott az akadályok leküzdéséhez, valamint a közös cél eléréséhez.