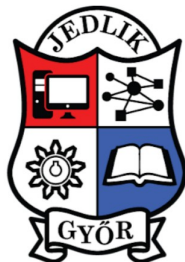




győri szakképzési centrum

Jedlik Ányos
Gépipari és Informatikai
Technikum és Kollégium



9021 Győr, Szent István út 7.

+36 (96) 529-480

+36 (96) 529-448

OM: 203037/003

jedlik@jedlik.eu

www.jedlik.eu

Záródolgozat feladatkiírás

Tanuló(k) neve: Somlói Dávid, Trifusz Huba, Verba Viktor
Képzés: nappali
Szak: 5 0613 12 03 Szoftverfejlesztő és -tesztelő technikus

A záródolgozat címe:

Elevate

Konzulens: Sándor László
Beadási határidő: 2025. 04. 15.

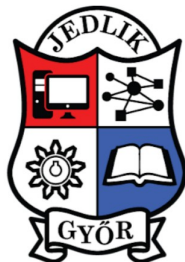
Győr, 2025. 02. 01

Módos Gábor
igazgató



győri szakképzési centrum

Jedlik Ányos
Gépipari és Informatikai
Technikum és Kollégium



9021 Győr, Szent István út 7.

+36 (96) 529-480

+36 (96) 529-448

OM: 203037/003

jedlik@jedlik.eu

www.jedlik.eu

Konzultációs lap

	A konzultáció		Konzulens aláírása
	ideje	témája	
1.	2025.02.15.	Témaválasztás és specifikáció	
2.	2025.03.14.	Záródolgozat készültségi fokának értékelése	
3.	2025.04.15.	Dokumentáció véglegesítése	

Tulajdonosi nyilatkozat

Ez a dolgozat a saját munkánk eredménye. Dolgozatunk azon részeit, melyeket más szerzők munkájából vettünk át, egyértelműen megjelöltük.

Ha kiderülne, hogy ez a nyilatkozat valótlan, tudomásul vesszük, hogy a szakmai vizsgabizottság a szakmai vizsgáról kizár minket és szakmai vizsgát csak új záródolgozat készítése után tehetünk.

Győr, 2025. április 15.

tanuló aláírása

tanuló aláírása

tanuló aláírása

Tartalomjegyzék

1	A projektről	4
1.1	Az Elevate célja	4
2	Weboldal	5
3	Mobil Applikáció	6
3.1	Technológia	6
3.2	Architektúra	6
3.3	Mappastruktúra	6
3.4	Főbb funkciók	7
3.5	Felhasználói élmény és dizájn	8
3.6	Natív integráció	8
3.7	Biztonság	8
3.8	Teljesítmény optimalizálás	8
4	Adatbázis	9
4.1	Adatbázis tervezés	9
4.2	Entitások és kapcsolatok	9
4.3	Adatbázis biztonsági megfontolások	10
4.4	Adatbázis elérés	10
5	Backend	11
5.1	Technológia	11
5.2	Architektúra	11
5.3	Végpontok	11
5.4	Autentikáció és biztonság	12
6	Tesztelés	14
7	Használati útmutató mobilhoz	15
7.1	Login	15
7.2	Szokás hozzáadás	16
7.3	Szokás teljesítés	17
7.4	Barátok hozzáadása	18
7.5	Baráti kérelem fogadás	19
7.6	Kihívás	20
7.7	Profilkép	22

1. A projektről

A téma kiválasztásánál arra törekedtünk, hogy egy, a hétköznapi élet során alkalmazható szoftvert készítsünk. Több opció is felmerült, azonban végül egy szokásformáló felület mellett döntöttünk, amit Elevate-nek neveztünk el, az egészséges, felemelő életmód jegyében. Az Elevate ösztönzi a felhasználókat, hogy új, pozitív szokásokat vezessenek be, miközben hatékonyan követhetik saját fejlődésüket, emellett hozzájárul életminőségük javításához és a fenntartható fejlődéshez.

1.1 Az Elevate célja

A szoftver célja, hogy a kliens az általa kívánt szokásokat fejlessze, vagy újakat építsen be a napirendjébe. Például, ha a felhasználó a dohányzásról szeretne leszokni, akkor monitorozni tudja a fogyasztását és különféle jutal-makat kap, ha tartja a felállított célját. Nem csak a rossz szokások követését biztosítja az applikáció, pozitív célokat is ki lehet tűzni, mint "Napi 10 fekvőtámasz" vagy "Hetente kitakarítani". Egy szokás tartásához elengedhetetlen, hogy a beállított gyakorisággal teljesítsük a kitűzött kihívásokat. Ennek megkönnyítése érdekében az Elevate egy naptárszerű nézetben jeleníti meg a teendőket és emlékeztet azok elvégzésére.

2. Weboldal

3. Mobil Applikáció

3.1 Technológia

Az Elevate mobilalkalmazása Ionic keretrendszerre épül, amely Angular alapú. Ez a kombináció lehetővé teszi a cross-platform fejlesztést, így egyetlen kódbázisból készíthető el az Android és iOS platformokra optimalizált alkalmazás. A felhasználói felület az Ionic komponenskönyvtárát és egyedi SCSS stílusokat használ. A backend API-val való kommunikáció Angular HttpClient-en keresztül történik.

3.2 Architektúra

Az alkalmazás a következő fő komponensekből épül fel:

- **Modulok** - Az alkalmazás funkcionális egységei
- **Komponensek** - Újrafelhasználható UI elemek
- **Oldalak** - Az alkalmazás különböző képernyői
- **Service-k** - Üzleti logika és adatkezelés
- **Modellek** - Az adatstruktúrák TypeScript interfészei
- **Stílusok** - SCSS fájlok a megjelenés testreszabásához
- **Capacitor plugin-ok** - Natív funkciók elérése (kamera, értesítések)

3.3 Mappastruktúra

Az Elevate mobilalkalmazás a következő mappastruktúrával rendelkezik:

Fő komponensek leírása:

- **components/** - Újrafelhasználható UI komponensek, amelyek több oldalon is megjelenhetnek (pl. szokás kártya, feed kártya)
- **models/** - TypeScript interfészek, amelyek az alkalmazásban használt adatstruktúrákat definiálják
- **pages/** - Az alkalmazás fő képernyői, minden képernyőhöz tartozik egy Angular komponens
- **services/** - A backend API-val való kommunikációt és egyéb adatkezelési funkciókat megvalósító szolgáltatások
- **guards/** - Útvonalvédelem, amely ellenőrzi a felhasználó jogosultságait az oldalak megtekintéséhez

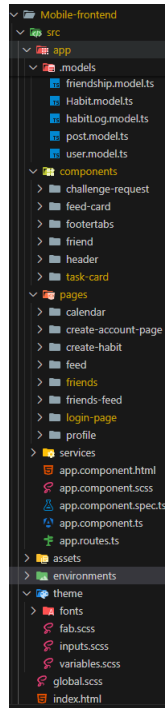


Figure 3.1: Mappastruktúra

3.4 Főbb funkciók

Az Elevate mobilalkalmazás a következő fő funkciókat kínálja:

Autentikáció:

- Felhasználói regisztráció validációval (jelszó erősség ellenőrzés)
- Bejelentkezés JWT token alapú hitelesítéssel
- Profilkezelés (profilkép feltöltése)

Szokáskövetés:

- Új szokások létrehozása
- Szokások személyre szabása (cím, leírás, szín, gyakoriság)
- Egyedi gyakoriság beállítása (napok kiválasztása)
- Napi szokások megjelenítése és teljesítésük követése
- Sorozatok (streak) nyilvántartása és vizualizálása

Feed és közösségi funkciók:

- Barátok tevékenységeinek követése
- Barátkérelmek kezelése
- Kihívások küldése és fogadása
- Felhasználók keresése

Naptár nézet:

- Aznap teljesítendő szokások követése
- Jövőbeli szokások előnézete

3.5 Felhasználói élmény és dizájn

Az alkalmazás felhasználói felülete a következő alapelvekre épül:

- **Reszponzív dizájn** - Alkalmazkodik különböző képernyőméretekhez
- **Sötét/világos téma** - Automatikus váltás a rendszerbeállítások alapján
- **Intuitív navigáció** - Alsó tabbar és oldalmenü kombinációja
- **Vizuális visszajelzések** - Animációk és toast üzenetek
- **Egyszerű űrlapok** - Validáció és hibaüzenetek

Az alkalmazás a Material Design elveit követi, egyedi színpalettával és tipográfiával kiegészítve. A fő színséma lila és kék árnyalatokra épül, ami a motivációt és a fejlődést szimbolizálja.

3.6 Natív integráció

A Capacitor segítségével az alkalmazás hozzáfér a készülék natív funkcióihoz:

- Kamera használata profilképek készítéséhez
- Eszköztéma-követés (sötét/világos mód)

3.7 Biztonság

Az alkalmazás biztonsági szempontjai:

- JWT token tárolása biztonságos módon
- Input validáció kliens oldalon
- Jelszavak biztonságos kezelése (minimális követelmények: 12 karakter, nagybetű, szám, speciális karakter)
- Nem autentikált felhasználók átirányítása a bejelentkezési oldalra

3.8 Teljesítmény optimalizálás

Az alkalmazás teljesítményét javító technikák:

- Lazy loading az oldalak betöltéséhez
- Infinite scroll a hosszú listák kezeléséhez
- Képek optimalizálása
- Standalone komponensek használata

4. Adatbázis

4.1 Adatbázis tervezés

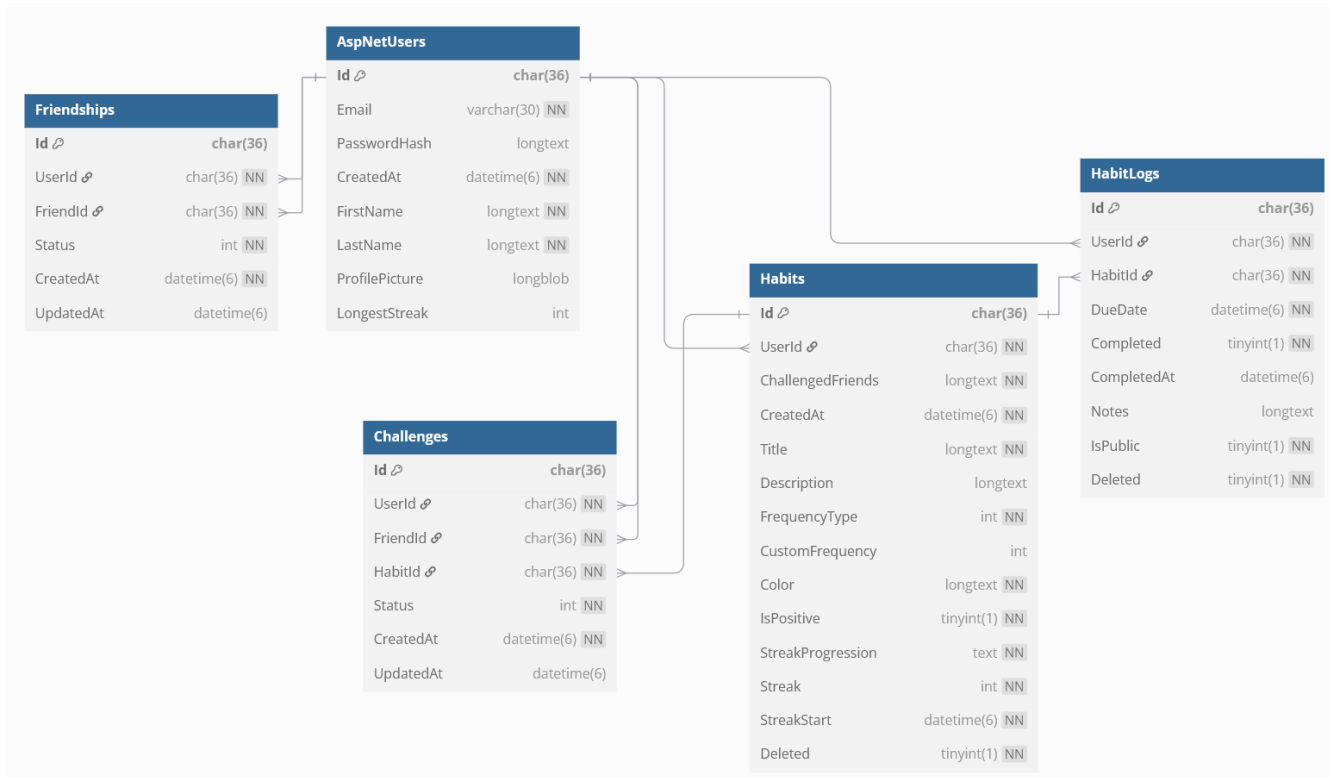
Az Elevate két különböző adatbázis rendszert támogat a különböző környezetekben való futtatáshoz:

- Fejlesztési környezetben: MySQL
- Production környezetben: PostgreSQL

Erre azért van így, mert a fejlesztést MySQL-el kezdtük, majd a Koyeb-re való publikáláshoz szükségessé vált PostgreSQL kompatibilitás. Az adatbázis migrációk kezelésére az Entity Framework Core migrációs rendszerét használjuk, amely lehetővé teszi a séma verziókövetését és az adatbázis automatikus frissítését, ez kisebb módosításokkal mindkét adatbázis környezettel megfelelően működik.

4.2 Entitások és kapcsolatok

Az adatbázis séma a következő fő táblákat tartalmazza:



4.3 Adatbázis biztonsági megfontolások

- A jelszavak hash-elve tárolódnak az adatbázisban (ASP.NET Core Identity)
- Adatbázis migrációk verziókövető rendszerben tárolva
- A kapcsolatok integritása constraint-ekkel biztosítva
- Indexek használata a gyakori lekérdezések optimalizálására

4.4 Adatbázis elérés

Az adatbázis elérését a `DbConnectionManager` osztály biztosítja az alábbi módon:

```
DbConnectionManager.cs

24 public DbConnection GetOpenConnection()
25 {
26     if (Environment.GetEnvironmentVariable("ASPNETCORE_ENVIRONMENT") ==
27         "Production")
28     {
29         var connection = new NpgsqlConnection(GetConnectionString());
30         connection.Open();
31         return connection;
32     }
33     else
34     {
35         var connection = new MySqlConnection(GetConnectionString());
36         connection.Open();
37         return connection;
38     }
39 }
```

Majd az így kapott kapcsolattal a `DbContext` osztály alkot egy, a későbbiekben feldolgozható adatszerkezetet.

```
ElevateDbContext.cs

61 protected override void OnModelCreating(ModelBuilder modelBuilder)
62 {
63     base.OnModelCreating(modelBuilder);
64
65     if (DbConnectionManager.IsProduction())
66     {
67         modelBuilder.UseIdentityAlwaysColumns();
68
69         modelBuilder.Entity<ApplicationUser>().ToTable("aspnetusers");
70         modelBuilder.Entity<IdentityRole<Guid>>().ToTable("aspnetroles");
71         modelBuilder.Entity<IdentityUserRole<Guid>>().ToTable("aspnetuserroles");
72         modelBuilder.Entity<IdentityUserClaim<Guid>>().ToTable("aspnetuserclaims");
73     }
74 }
```

5. Backend

5.1 Technológia

Az Elevate backend rendszere ASP.NET Core alapú, Entity Framework Core ORM-mel. Az adatbázis és a backend kapcsolata model first elv alapján lett létrehozva. Az API RESTful elvek alapján lett kialakítva és a CRUD (Create, Read, Update, Delete) műveleteket valósítja meg.

5.2 Architektúra

A backend a következő komponensekből épül fel:

- **Modellek** - Az adatmodelleket és adatbázis entitásokat reprezentálják
- **DTO-k (Data Transfer Objects)** - Adatok átvitelére szolgáló objektumok a rétegek között, illetve a kliens és szerver között
- **Repository-k** - Az adatbázissal való kommunikációért felelősek, CRUD műveletek végrehajtása
- **Kontrollerek** - A kérések feldolgozása, autentikáció és autorizáció kezelése, valamint a válaszok generálása
- **Service-k** - Az üzleti logika megvalósítása
- **Middleware** - Kivételek kezelése és egyéb előfeldolgozási feladatok
- **Segédosztályok** - Általános funkciók és segédszolgáltatások

5.3 Végpontok

A Backend API részletes dokumentációja a [Swagger](#) felületen érhető el. Az alábbiakban a főbb végpontok láthatóak:

- **Autentikáció**
 - Regisztráció (**POST** /api/auth/register)
 - Bejelentkezés (**POST** /api/auth/login)
- **Felhasználó**
 - Felhasználó adatainak lekérése email alapján (**GET** /api/user)
 - Felhasználó adatainak lekérése id alapján (**GET** /api/user/:id)

- Felhasználó adatainak frissítése (**PATCH** /api/user/:id)
- Szokások
 - Szokások listázása (**GET** /api/habit)
 - Szokás lekérése azonosító alapján (**GET** /api/habit/:id)
 - Új szokás létrehozása (**POST** /api/habit)
 - Szokás módosítása (**PATCH** /api/habit/:id)
 - Szokás törlése (**DELETE** /api/habit/:id)
- Szokás napló
 - Szokás naplók listázása (**GET** /api/habitlog)
 - Szokás napló lekérése azonosító alapján (**GET** /api/habitlog/:id)
 - Napi szokás naplók lekérése (**GET** /api/habitlog/:dueDate)
 - Szokás napló frissítése (**PATCH** /api/habitlog/:id)
- Kihívások
 - Kihívások lekérése felhasználó azonosító alapján (**GET** /api/challenge/:userId/challenges)
 - Kihívás meghívók listázása (**GET** /api/challenge/:userId/challenge-invites)
 - Elküldött kihívás meghívók listázása (**GET** /api/challenge/:userId/sent-challenge-invites)
 - Új kihívás létrehozása (**POST** /api/challenge)
 - Kihívás státuszának frissítése (**PATCH** /api/challenge)
 - Kihívás törlése (**DELETE** /api/challenge)
- Feed
 - Feed bejegyzések lekérése (**GET** /api/feed)
- Barátok
 - Barátok listázása (**GET** /api/friendship/:userId/friends)
 - Beérkezett barátkérések lekérése (**GET** /api/friendship/:userId/friend-requests)
 - Küldött barátkérések lekérése (**GET** /api/friendship/:userId/friend-requests-sent)
 - Barátkérés küldése (**POST** /api/friendship)
 - Barátkérés elfogadása/elutasítása (**PATCH** /api/friendship)
 - Barátság törlése (**DELETE** /api/friendship)

5.4 Autentikáció és biztonság

Az API biztonságos használatához JWT (JSON Web Token) alapú autentikáció van implementálva. A működése:

- A felhasználó bejelentkezéskor egy JWT token-t kap (aszimmetrikus titkosítással)

- A token érvényességi ideje korlátozott
- A védett végpontok eléréséhez a token minden kérés fejlécében el kell küldeni

A biztonság további rétegei:

- Input validáció
- CORS védelem (A mobil alkalmazás miatt enyhített)
- Jelszó titkosítás

6. Tesztelés

End to end tesztelés Cypress használatával:

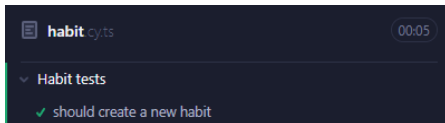


Figure 6.1: Első kép

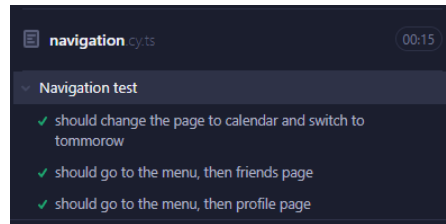


Figure 6.2: Második kép

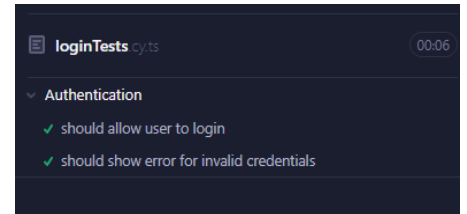


Figure 6.3: Harmadik kép

A fenti képeken a Cypress által végrehajtott end-to-end tesztek láthatók, amelyek ellenőrzik az alkalmazás különböző funkcióinak működését. A tesztelés során ellenőriztük a felhasználói bejelentkezést, az adatbevitel validációját és az egyes oldalak közötti navigációt is.

7. Használati útmutató mobilhoz

7.1 Login

A következő képernyőképek a mobilos bejelentkezési folyamat lépéseit mutatják be:

1. lépés

2. lépés

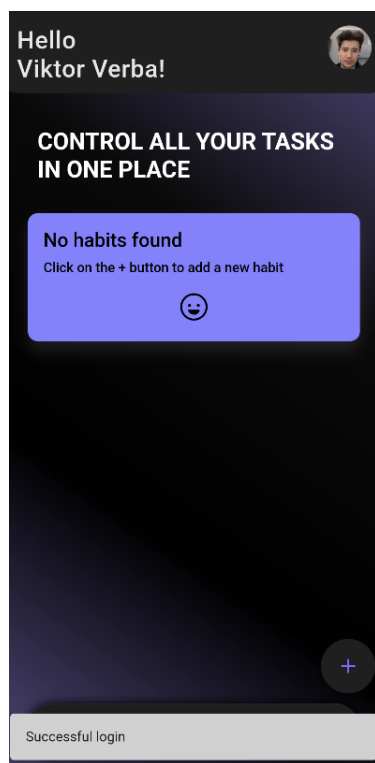
3. lépés

4. lépés

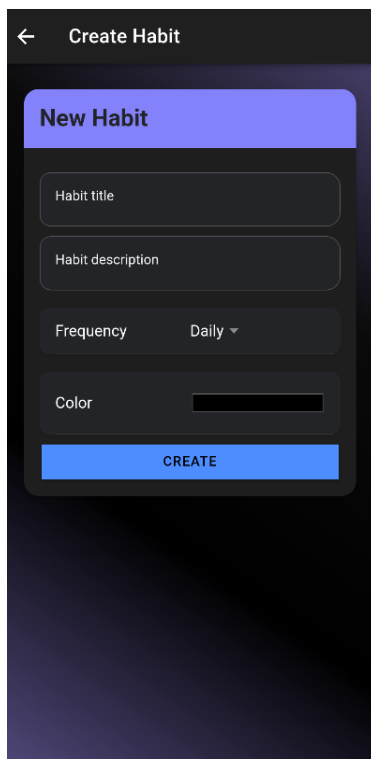
Mobilos bejelentkezés lépései

Amennyiben a felhasználó nem rendelkezik fiókkal, meg kell nyomnia a Create one here! gombot. A sikeres regisztráció után automatikusan átirányítás történik a bejelentkezési oldalra, ahogy be kell jelentkezni a már meglévő fiókkal. Amennyiben a felhasználó rendelkezik fiókkal, nem kell regisztrálnia. Sikeres bejelentkezés után automatikusan átirányítás történik a Szokások (Habits) oldalra

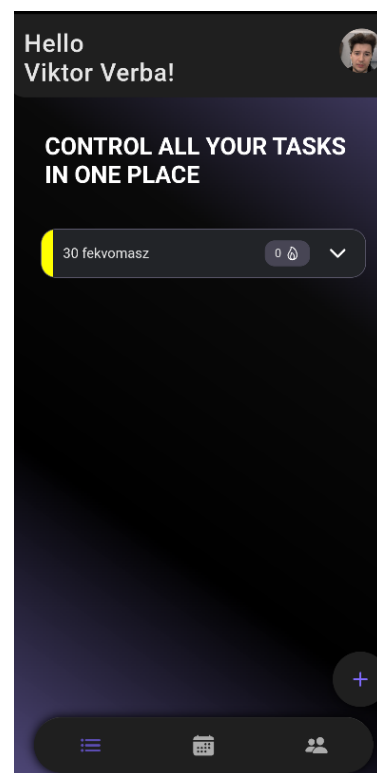
7.2 Szokás hozzáadás



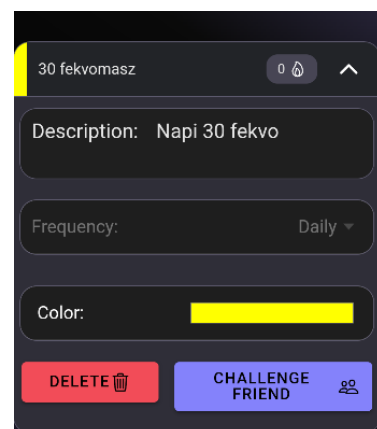
Szokás hozzáadása, meg kell nyomni a + gombot



Szokás adatainak feltöltése



Láthatjuk a frissen hozzáadott szokást

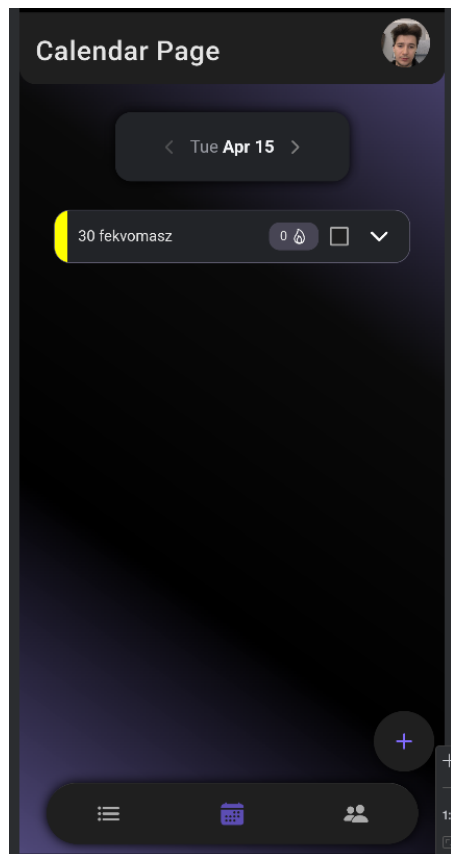


Kattintásra kinyílik a szokás

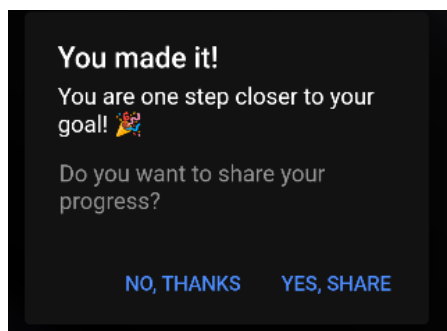
Szokás létrehozásának lépései a mobilalkalmazásban

A fenti képeken látható, hogyan lehet egy új szokást hozzáadni az alkalmazáson belül. A felhasználó kiválasztja a szokás típusát, megadja a nevet, gyakoriságot, majd elmenti azt. Ez segít abban, hogy a napi célkitűzések következetesen teljesüljenek.

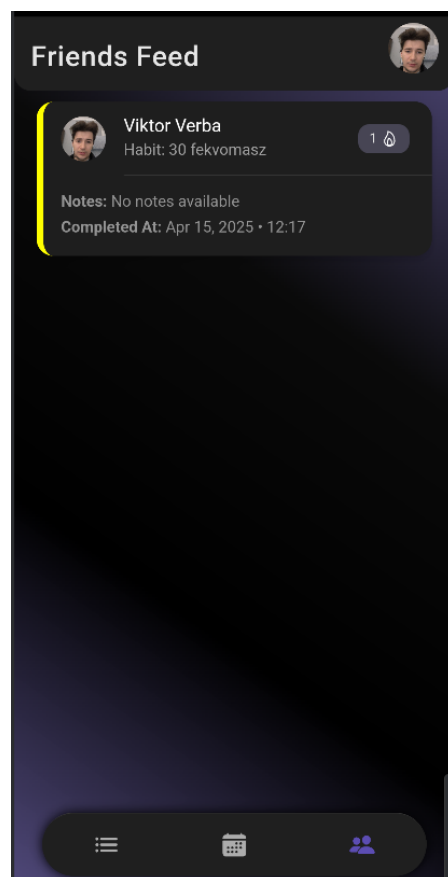
7.3 Szokás teljesítés



Napi nézett, kis kockára kattintva teljesíthető a szokás



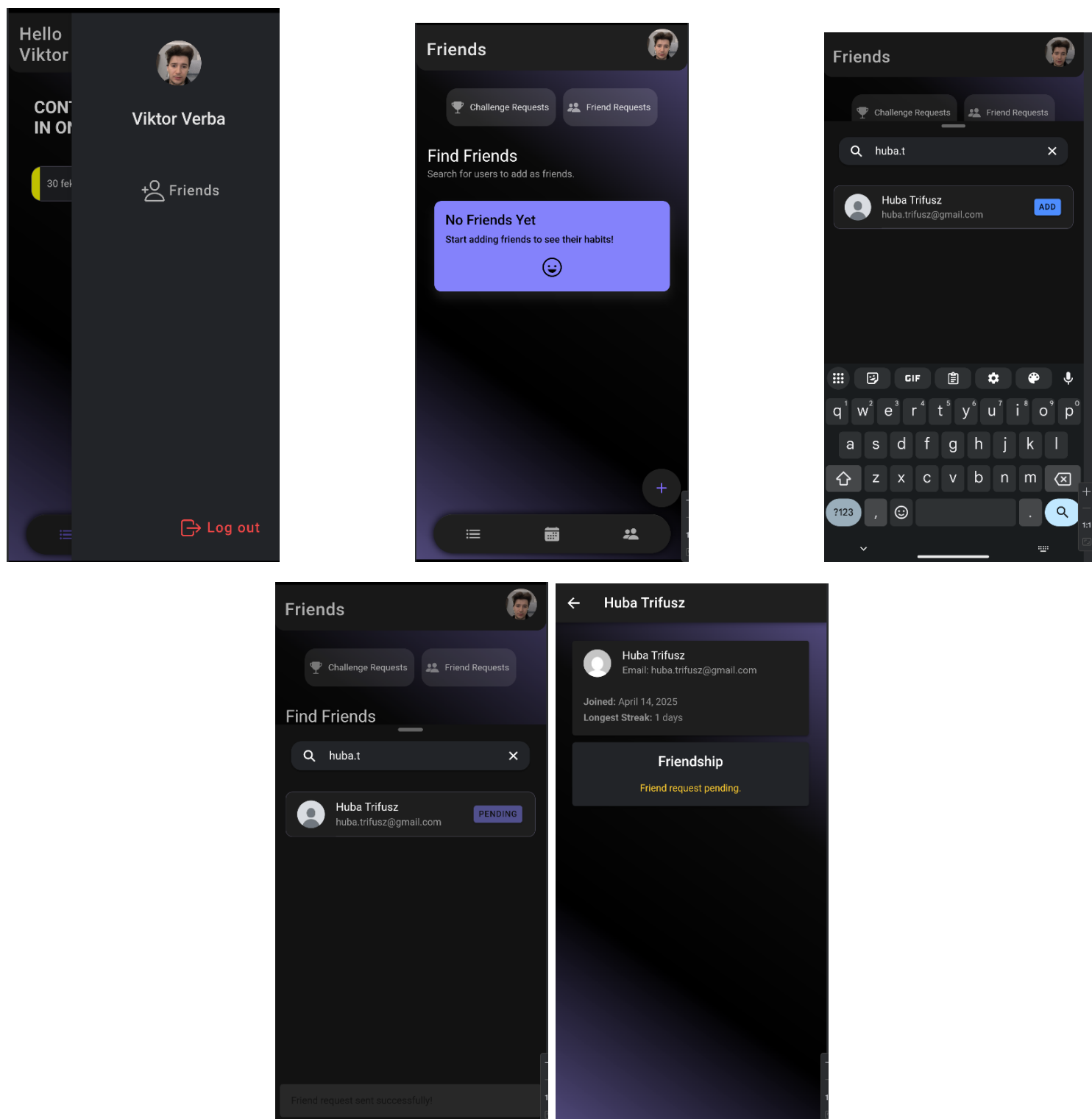
Kiválasztás hogy publikus legyen-e a teljesítésünk



Amennyiben publikusat választottunk, a szokás megjelenik a feed oldalon

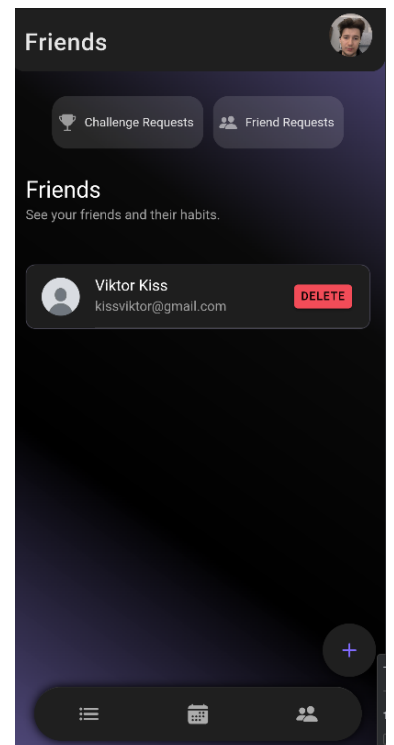
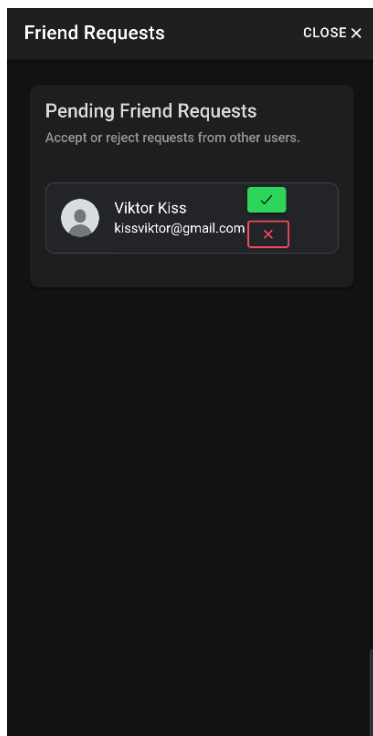
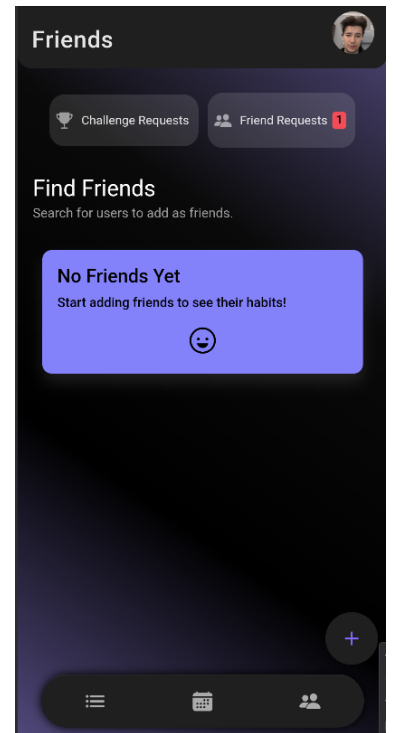
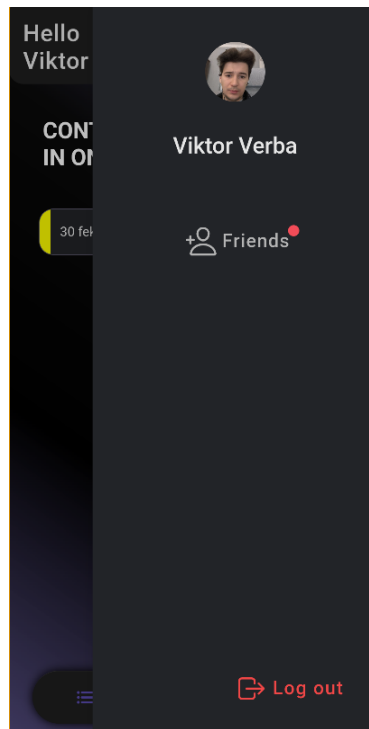
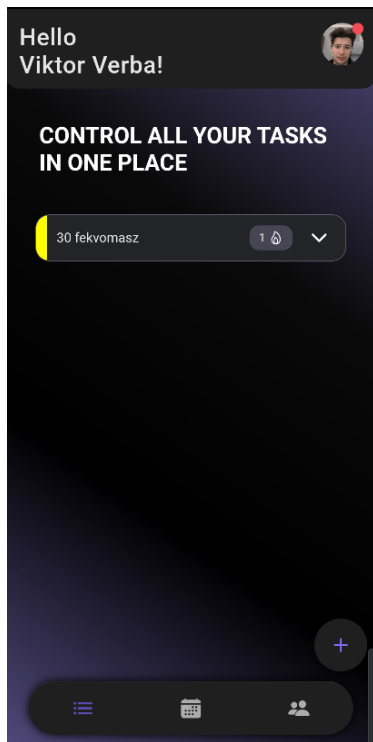
A fenti lépések bemutatják, hogyan lehet egy szokást teljesítettként jelölni a naptár nézetben és megjeleníteni feed-ben. Kalendár nézetben válthatjuk a napokat nyílak segítségével.

7.4 Barátok hozzáadása



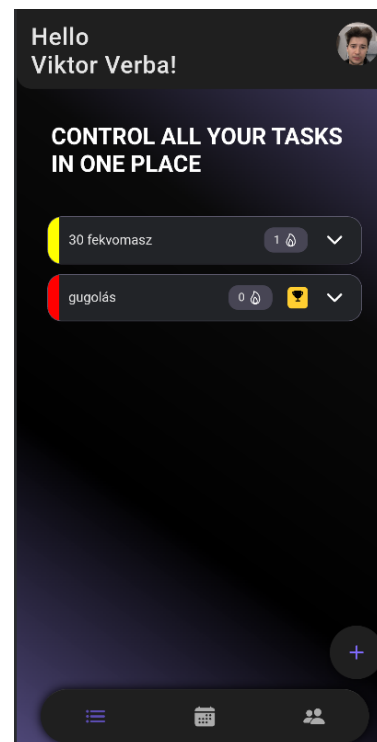
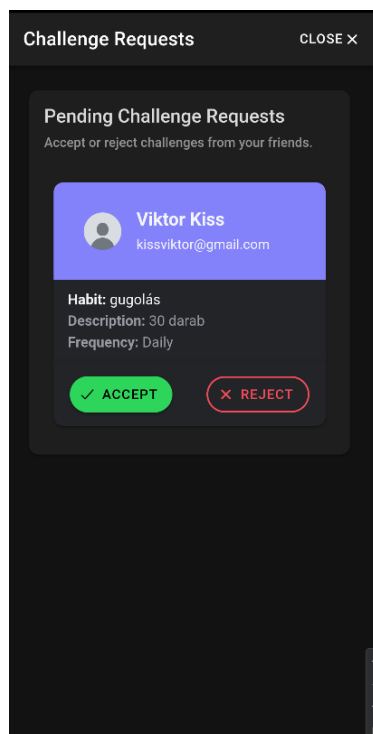
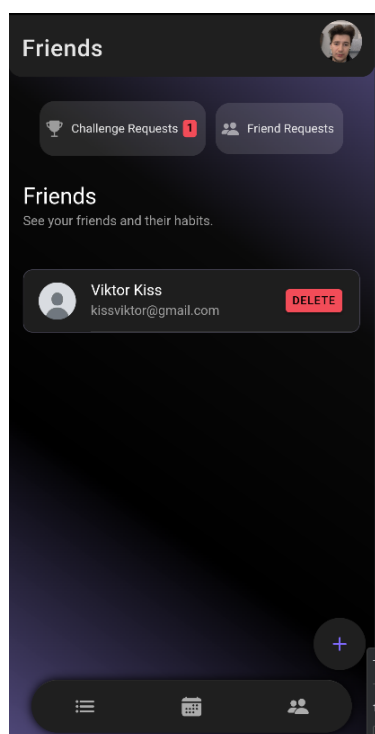
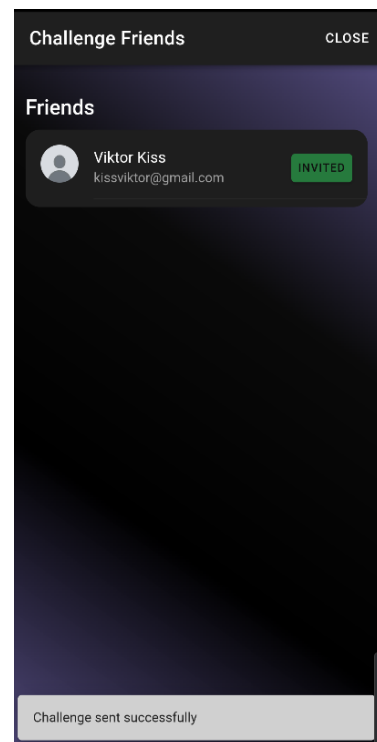
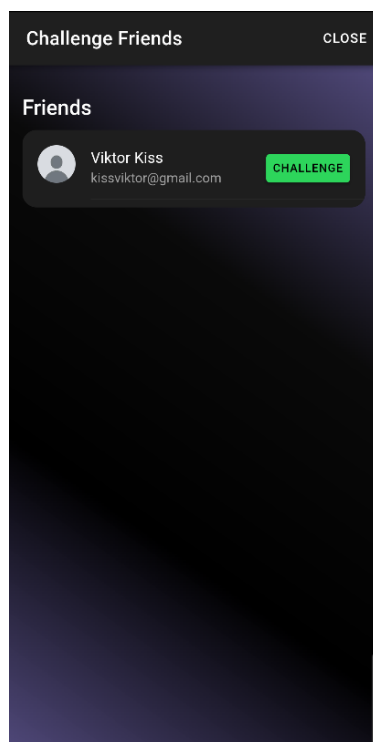
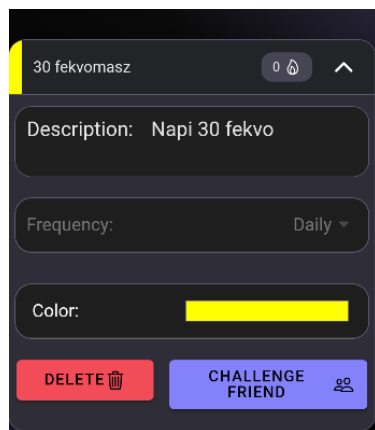
Profilképre kattintva megjelenik a menü, ahogy Friends gombra kattintva átírányít minket a program Barátok oldalra. Itt láthatjuk hogy nincs barátunk, + gombra kattintva kereshetünk a felhasználók között. Miután megtaláltunk a keresett személyt rákattinthatunk az Add gombra. Miután megnyomtuk a gomb felírata megváltozik. Keresett felhasználó profilképére kattintva megtekinthetjük a felhasználó összes adatát és azt is hogy a baráti kérelem el lett küldve.

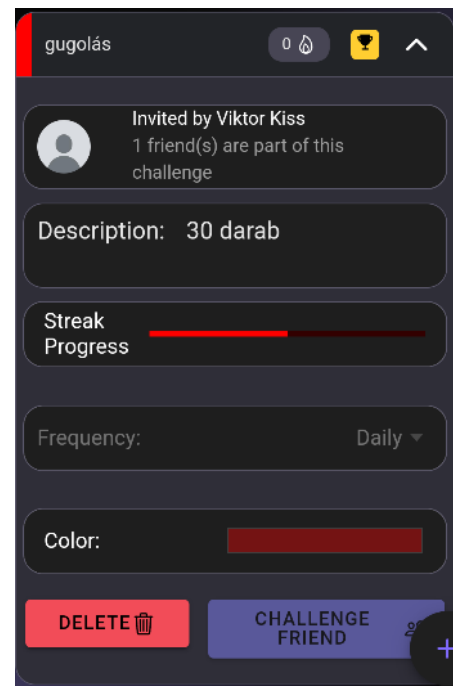
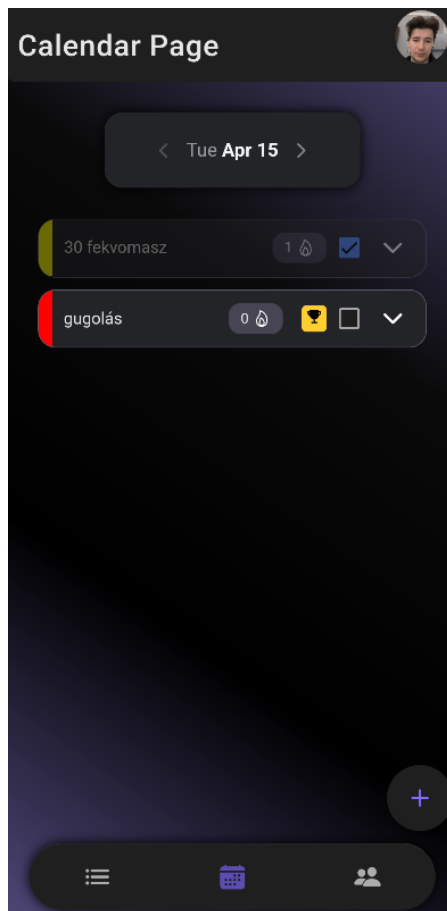
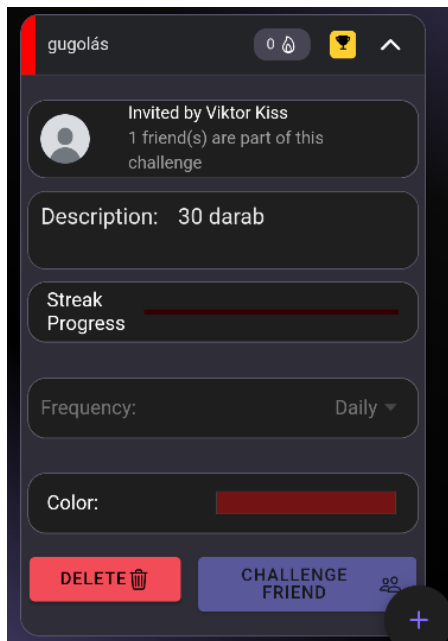
7.5 Baráti kérelem fogadás



Amennyiben a program észlel, hogy valaki küldött nekünk baráti kérelmet vagy challenge kérelmet a profilképnél megjelenik egy kis piros pötty. Miután rákattintunk a profilképünkre a Friends gombon is látni fogjuk a pöttyöt, ami azt jelzi, hogy ezen az oldalon érkezett kérelem. Miután barátok oldalra lépünk láthatjuk a baráti kérelem és challenge gombokat. Piros szám azt jelzi hogy hány kérelem érkezett hozzánk. Friend Requests gombra kattintva megjelennek a beérkezett baráti kérelmek. Elfogadás után a barátunk megjelenik a listában.

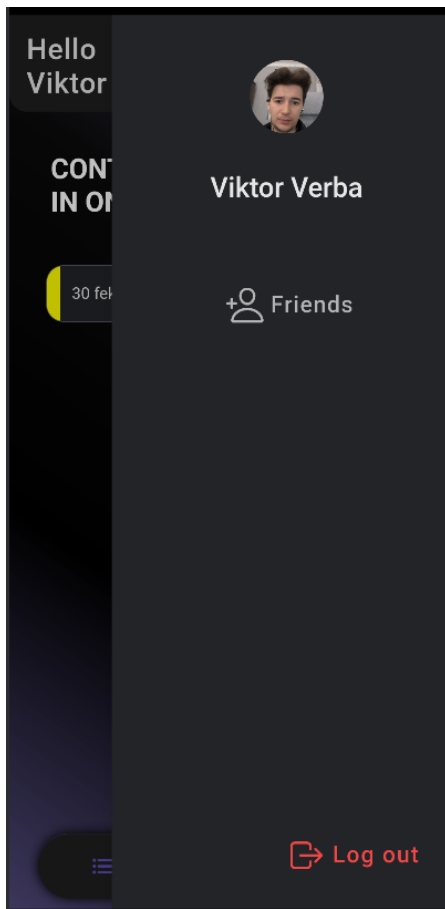
7.6 Kihívás



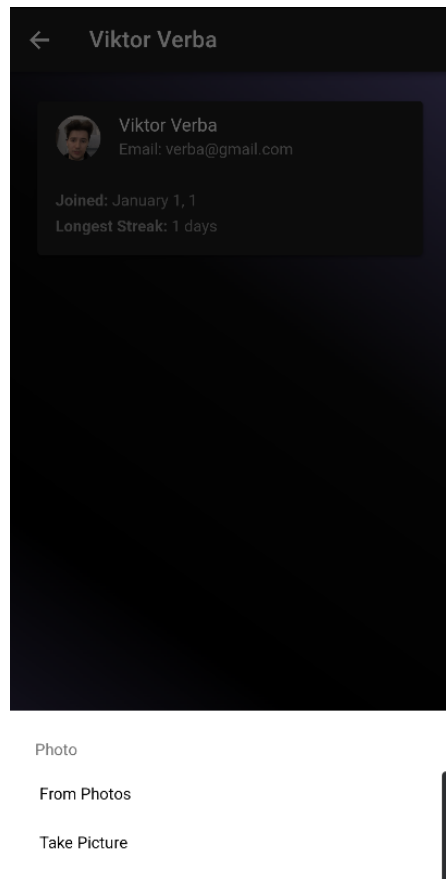


Amennyiben a felhasználó rendelkezik barátokkal, akkor tud küldeni a barátainak kihívást (challenget) amit onnantól közösen kell elvégezniük ahhoz, hogy a streak nőjön. Szokásra kattintva rá kell kattintani a Challenge friend gombra, amely átvész egy oldalra, ahol kiválaszthatjuk a barátot, akit meg akarjuk hívni. Miután rákattintottunk a challenge gombra, kapunk visszajelzést és a gomb átváltozik Invited-re. További képeken látható, hogy kell elfogadni vagy elutasítani a kihívásokat. Amennyiben elfogadjuk a kihívást az meg fog jelenni a szokásaink közül. Kihívást egy kis ikonnal tudunk megkülönböztetni. Ha valaki meghívott minket, akkor láthatjuk ki volt az és hányan vesznek részt benne. Streak Progress mutatja, hogy eddig hányan teljesítették a résztvevők közül a kihívást.

7.7 Profilkép



Profilképre kattintva átvissz minket a profil oldalra.



Profilképre kattintva kiválaszthatjuk, hogy képek közül akarunk választani vagy csinálni akarunk egy képet.



Képcsinálásra megnyílik a kamera