

**EÖTVÖS LORÁND TUDOMÁNYEGYETEM  
INFORMATIKAI KAR**

Komputeralgebra  
Tanszék

Peer-to-Peer architektúrán alapuló apróhirdetés oldal

Témavezető: Szerző:

**Kovács Attila Szabó Áron**

egyetemi tanár Programtervező informatikus BS, nappali tagozat

Budapest, 2022

***Tartalomjegyzék***

[1. Bevezetés 3](#_Toc104518218)

[1.1. Témabejelentő 3](#_Toc104518219)

[1.2. Felvezetés 3](#_Toc104518220)

[1.3. Téma 4](#_Toc104518221)

[1.4. Zero Trust User Environment 4](#_Toc104518222)

[2. Felhasználói dokumentáció 5](#_Toc104518223)

[2.1. Alkalmazás használatához szükséges követelmények 5](#_Toc104518224)

[2.1.1. Minimum rendszerkövetelmények 6](#_Toc104518225)

[2.1.2. Ajánlott rendszerkövetelmények 6](#_Toc104518226)

[2.2. Alkalmazás telepítése 7](#_Toc104518227)

[2.2.1. Az adatbázis telepítése, létrehozása és futtatása 7](#_Toc104518228)

[2.2.2. Adattároló szerver telepítése, létrehozása és futtatása 8](#_Toc104518229)

[2.3. Felhasználói felület bemutatása 9](#_Toc104518230)

[2.3.1. Köszöntő oldal és főmenü 9](#_Toc104518231)

[2.3.2. Elérhető szerverek megtekintése 11](#_Toc104518232)

[2.3.3. Hirdetések megtekintése 12](#_Toc104518233)

[2.3.4. Profil kezelése 13](#_Toc104518234)

[2.3.5. Még több információ a P2P-PP projektről 13](#_Toc104518235)

[3. Fejlesztői dokumentáció 14](#_Toc104518236)

[3.1. Követelményspecifikáció 14](#_Toc104518237)

[3.2. Környezet leírása 15](#_Toc104518238)

[3.3. Felhasznált technológiák 15](#_Toc104518239)

[3.3.1. Architektúra 15](#_Toc104518240)

[4. Továbbfejlesztési lehetőségek 16](#_Toc104518241)

[5. Összefoglalás 17](#_Toc104518242)

[6. Köszönetnyilvánítás 18](#_Toc104518243)

[7. Irodalomjegyzék 19](#_Toc104518244)

[8. Mellékletek 20](#_Toc104518245)

1. Bevezetés
   1. Témabejelentő

A cél egy olyan webes applikáció keretrendszerének megtervezése és megvalósítása, amely decentralizált aggregáló szerverek segítségével megengedni egy hétköznapi apróhirdetéses weboldal peer-to-peer megvalósítását. Kulcsfontosságú kihívás, hogy a peer-to-peer rendszer kliens oldalon a HTTPS weben egyszerűen, külön felhasználó oldali „overhead” nélkül elérhető ÉS átlátható legyen. Az egyes felhasználói profilok központi adatbázisban való tárolása helyett a fájlok megosztottak, amelyekben az egyes felhasználóhoz kötött apróhirdetések egyszerű, átlátható formában vannak jelen.

1. Kliens oldal: egy első szintű, normál (vég-)felhasználó esetén lényegében nem lenne különbség a hagyományos modellhez képest. Egy második szintű (dev) felhasználó saját szerver helyi hosztolásával tud hozzájárulni a hálózat bővítéséhez/fenntartásához.
2. Protokoll: A fájlokat a felhasználók egymás és az aggregáló szerverek között szabványos internetes protokoll segítségével osztják meg.
3. Aggregáló szerverek: mivel adott protokoll szigorúan P2P kapcsolatok kiépítésével működik, ezért a folyamatosan fenntartott szinkronizáló és aggregáló szerverek szükségesek, ahol egyes felhasználók biztonságosan tárolhatják profiljaikat (fájljaikat) és szinkronban tarthatják azokat.
   1. Felvezetés

Napjainkban nyilvánvaló igazságnak bizonyul, hogy az általunk ismert internet jelenlegi formájában tarthatatlan állapot felé közelít, hacsak nem érte már el azt. Egy újabb korszak eljövetelét követeli, mely felépítésében alapként szolgál az egyéni felhasználó szükségszerű magánjogainak és opcionális anonimitásának szabadon történő irányítása. Az internet korában született generációk feltehetőleg figyelmesebben és mélyebb megértéssel fogják azt használni, mint az őket megelőzők, így az általuk való felhasználásra új dizájnú weboldalak is kellenek majd.

* 1. Téma

A Peer-to-Peer (vagy P2P) Piacportál névre hallgató dolgozatom témája egy ilyen újhullámú weboldalt – és a mellé járuló minimalista keretrendszert – hivatott megvalósítani. Alapként egy apróhirdetés platform szolgál, hiszen a *„cserekereskedelem”* egyike társadalmunk alappilléreinek, a magántulajdon mellett, és az ilyen jellegű szolgáltatások csak kibővítik és felhatalmazzák felhasználóikat ezen szabadságukkal való élésre. A felhasználók egy letisztult felületen keresztül férhetnek hozzá egy alapszolgáltatásokat biztosító aggregátor szerverhez. Az aggregátor szerver szerepe, hogy több, kisebb, egymással egyenlő adattároló szervereket gyűjtenek össze, hogy azokat könnyű legyen átlátni, tartalmukat böngészni. Identitásukat RSA kulcspár aláírásának segítségével tudják saját posztolásaikhoz kötni és a szervernek azonosítani azt.

* 1. Zero Trust User Environment

A webszolgáltatás egyik kulcsfontosságú eleme a sajátos *„Zero Trust User Environment”* azaz Teljes Megbízhatatlansági Felhasználói Környezet. Ez egy meglévő és alkalmazott kiberbiztonsági fogalom átgondolása. Kiindulásként azzal a feltételezéssel él, hogy a rendszer nem megbízható, így minden végpont, minden felhasználó minden felhasználási eset egy szabálysértést végez, ezért minimális jogosultságú és hozzáférést ad. A P2P Piacportál (továbbiakban P2P-PP) ezt a mentalitást kiterjeszti a rendszerről a felhasználóra, így azok egyéni adatait saját gépükön tárolja, mely adatok biztonságát a felhasználó garantálhatja. Továbbá metaadatokat nem használ, sütiket nem tárol, harmad-félre analitikának helyet nem ad. Azaz, lehetővé teszi, hogy minél szabadabban, letisztultan és biztonságosabban végezzék a hétköznapi emberek apróhirdetéseik, cserekereskedelmük lebonyolítását.

1. Felhasználói dokumentáció

Az architektúra kialakításának érdekében fontos, hogy a különböző típusú felhasználókat jól el lehessen különíteni egymástól. Mivel a két számontartott felhasználási mód jelentősen eltér egymástól meg kell különböztetnünk két külön szerepkört, mely jelen esetben a kliens szerepét betöltő végfelhasználó és a rendszert tovább bővítő, saját adattároló szervert fenntartó, dev felhasználó.

Előbbi a weben keresztül csatlakozik a *„backend”*-ként funkcionálni aggregátor szerverhez csatolt klienshez/*„frontend”*-hez. Számára a weboldalon keresztül történik kizárólag az alkalmazással az interakció. Ott tudja profilját kezelni, regisztrációt követően bejelentkezni, hirdetést létrehozni, a már meglévői hirdetéseit megtekinteni, módosítani vagy törölni. Továbbá, regisztráció nélkül is lehetséges az oldal használata, hiszen már feltöltött hirdetéseket és az aktuálisan elérhető szervereket bárki böngészheti az ennek szánt felületeken.

Az utóbbi szerepkör saját szerver kihelyezésével bővíti a rendszert. Ebből következően a számára releváns kontakt felület a parancssor. Feltéve, hogy megfelel a követelményeknek és beszerzi a szükséges szerverfájlokat, a második szintű felhasználó parancssorból kell beállítsa a megfelelő környezeti változókat, frissíteni/telepíteni a szükséges modulokat és elindítani a szervert.

A két végletet, pedig, egy aggregátor szerver köt össze. A három hálózati pont egy kísérleti módszer segítségével kommunikál egymással, standard internet protokollt használva. A végfelhasználó kéréseit az aggregátor irányítja a haladó felhasználók által futtatott adattároló szerverek felé.

* 1. Alkalmazás használatához szükséges követelmények

Mivel az alkalmazás kliens oldala a webre készült, platform független, bármilyen modern operációs rendszerről elérhető alkalmas böngésző használatával. Viszont egy szerver kitelepítéséhez kellenek az ehhez szükséges keretrendszerek és segédprogramok, modulok. A kliens- és szerveroldali fejlesztés egyaránt JavaScript ES6 nyelvnek a TypeScript túlterhelésével történt. A Node.js v17.8+ környezet szükségszerű a szerver futtatásához. Az adatok a szerveren helyileg vannak tárolva, melyet az SQLite3 v4.0.23+ keretrendszerben a Prisma2 ORM v3.11+ segítségével. Ezek telepítése elengedhetetlen.

* + 1. Minimum rendszerkövetelmények

A fentiek alapján az alábbi operációs rendszerek vannak támogatva:

* Windows 10 21H2+
* Windows 11 22H2+
* Windows Server 2012 R2 (64-bit)
* Mac OS X 12.0+
* Ubuntu Linux 20.04+

Mivel a szoftverkomponensek hardveres erőforrásigénye napjainkban elhanyagolható, a minimális hardverkövetelmények meghatározásakor a választott operációs rendszer mérvadó. A használt keretrendszerek követelményeinek a következő böngészők tesznek eleget:

* Google Chrome/Chromium 100+
* Safari 12
* Microsoft Edge 41+
* Opera 56.0

Továbbá megjegyzendő, hogy az alkalmazás működéséhez a folyamatos internetkapcsolat fenntartása szükséges. Természetesen a támogatott internetböngészőkön kívül mással is megnyitható a weboldal, de az összes funkció hibátlan működése nem garantált.

* + 1. Ajánlott rendszerkövetelmények

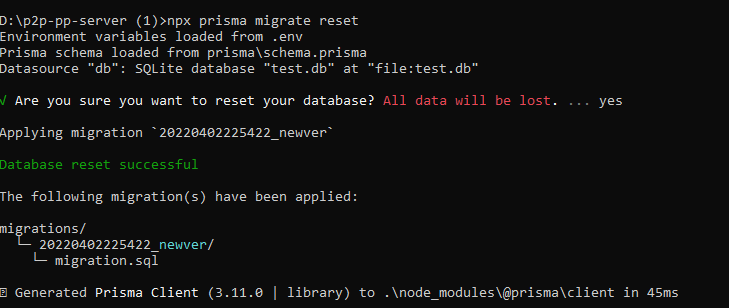
A gördülékeny felhasználás érdekében ajánlott a kliensoldali megjelenítéshez Vivaldi 5.2.2623.48 (Chromium 100) vagy modernebb böngésző. Továbbá javasolt egy 16:9 vagy 21:9 képarányú, legalább FullHD (1920\*1080 vagy 2560\*1080 felbontású) képernyő használata. Hasonlóan ajánlott a fejlesztéshez használt operációs rendszer, Windows 11 Pro 22H2 x64 build 22621.1.

* 1. Alkalmazás telepítése

A weboldal eléréséhez, természetesen, nincs szükség semmilyen specializálódott szoftvert telepítéséhez. Adattároláshoz szervereket azonban szükséges telepíteni. Az alkalmazás telepítését Windows 11 operációs rendszerre ajánlott, de mivel a felhasznált technológiák platformfüggetlenek, a megfelelő függőségek installálása után feltételezhetően más rendszereken is hasonlóképpen működik.

* + 1. Az adatbázis telepítése, létrehozása és futtatása

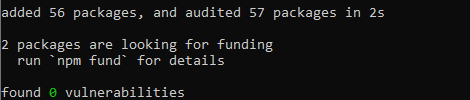
Az adatbázis, mint fogalom szét van osztva az összes dev felhasználó által működtetett szerver között. Az adatok tárolása a szerveren történik, helyi fájlban vannak elmentve a hirdetések és azok kategóriái, SQLite3 adatbázis motor könyvtár segítségével. Az adatbázis telepítése, létrehozása, futtatása és karbantartása a szerver program beépített funkciói közé tartoznak, így a dev felhasználó csak az előre létező függőségek biztosításáról kell gondoskodjon. Teljesen új szerver telepítése esetén a szerverfájlok/függőségek telepítése után, de annak elindítása előtt kell biztosítani a Prisma ORM kliens sikeres létrejöttét. Ezt a npx prisma migrate reset majd a npx prisma db push parancsok lefuttatásával lehet elérni. Ha az sikeresen megtörtént a parancssorban hasonló üzenet kéne fogadjon:



1. ábra - adatbázis létrehozása

* + 1. Adattároló szerver telepítése, létrehozása és futtatása

Az adattárolásra alkalmas szervereket a második szintű felhasználók is képesek működtetni. Ennek első lépése letölteni a szerverhez szükséges fájlokat. Utána lehet figyelembe venni a szükségszerű függőségek telepítését. Mivel a server JavaScript ES6 nyelvre van építve, ezért nem szükséges nyelv vagy környezet letöltése, de a szerver futtatásához szükséges a Node.js futásidejű keretrendszer legalább 17.8-as verziója. Ezt a Node.js hivatalos oldaláról lehet beszerezni (<https://nodejs.org/en/download/>). A telepítővel automatikusan installálódik a Node.js csomagkezelője, az npm (Node Package Manager) is, melynek segítségével gondoskodhatunk az összes többi függőség automatikus telepítéséről. A sikeres telepítésről, illetve a megfelelő verziószámról a node --version parancssori parancs lefuttatásával tudunk megbizonyosodni. Amennyiben a Node Package Manager sikeresen fel lett telepítve az npm install --omit=dev paranccsal lehet a *package.json* fájlban feltűntetett függőségeket automatikusan telepíteni. Miután ez sikeresen megtörtént hasonló üzenetet kell kapjunk:

Ezután biztosítanunk kell a szerver helyes működése érdekében az adatbázis létrehozását[[1]](#footnote-2). Továbbá, bár beépített biztonsági hálóknak köszönhetően nem szükségszerű, a helyes és elvárt működéshez kimondottan ajánlott beállítani a futási környezet változóit (portszám, aggregátor elérhetősége, saját elérhetőség) az alábbi parancsokkal:

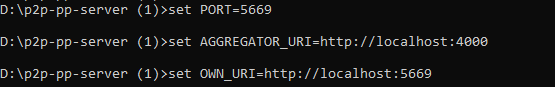
2. ábra - npm install eredménye

set PORT={használt portszám}

set AGGREGATOR\_URI={aggregátor link}

set OWN\_URI={saját link}

Erre egy alkalmas példa:



3. ábra - környezeti változók beállítása

* 1. Felhasználói felület bemutatása
     1. Köszöntő oldal és főmenü

Az alkalmazás kliens oldala egy böngészőből elérhető, a kihelyezett frontend címét beírva az URL sávba (jelen esetben ez a localhost lesz). A megnyitást követően a bejelentkező oldalon találjuk magunkat.

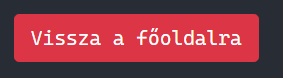


4. ábra – A köszöntő oldal

A lefele mutató gombra kattintva (vagy csak hagyományosan legörgetve az egérrel) megtekinthetjük a főmenüt, ahol kiválaszthatjuk, hogy a szervereket szeretnénk böngészni vagy a profilokat, esetleg a saját felhasználói profilunkkal szeretnénk foglalkozni vagy több infót akarunk olvasni a P2P-PP projektről:



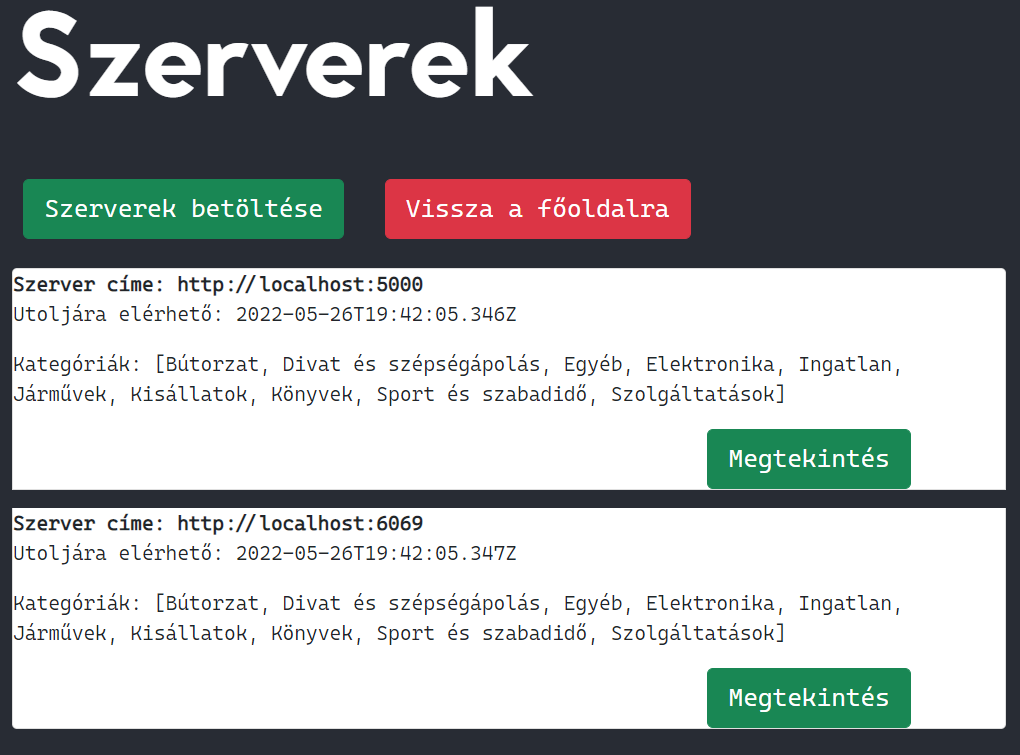
5. ábra - A főmenü négy gombja

Ezek a menüpontok megjelenési sorrendben lesznek részletesebben szemléltetve. A köszöntő oldalt/főoldalt leszámítva minden egyes oldalon megtalálható egy *Vissza a főoldalra* feliratú gomb, mely, értelemszerűen, bárhonnan visszairányít minket a főmenühöz.

6. ábra - Főoldalra visszairányító gomb

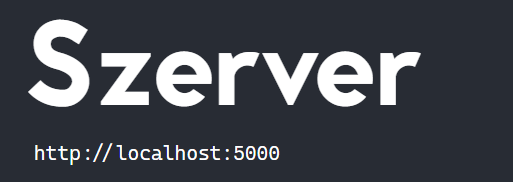
* + 1. Elérhető szerverek megtekintése

Az első gombra kattintva a főmenüben át leszünk irányítva az oldalra, ahol listába rendezve böngészhetjük az aggregátorhoz aktuálisan csatlakozott szervereket. Az oldalon két gomb található, a korábban leírt Vissza gomb és a *Szerverek betöltése*-ért felelős gomb, mely be- vagy újratölti az aggregátor által nyilvántartott szerverek listáját. Az oldalra érkezéskor ezek automatikusan be vannak töltve, de a gomb használatával bizonyosodhatunk, hogy a lehető legfrissebb változatát látjuk a listának.

A szerverek adatai listaelemenként uniform formátumban jelennek meg, fehér háttérrel. Feliratozva elemenként láthatjuk adott szerver elérési címét, a számontartott kategóriáit és az utolsó biztosan aktív dátumát. Innen a *Megtekintés* gombbal lehet az egyéni adattárolón lévő hirdetéseket megtekinteni.

Ezeket a szerver specifikus hirdetéseket a szokásoshoz[[2]](#footnote-3) hasonló módon jeleníti meg. Egyetlen jelentős eltérés az az oldal elején található, hiszen jelen esetben fel van tüntetve a megtekintve lévő szerver elérhetőségi címe.

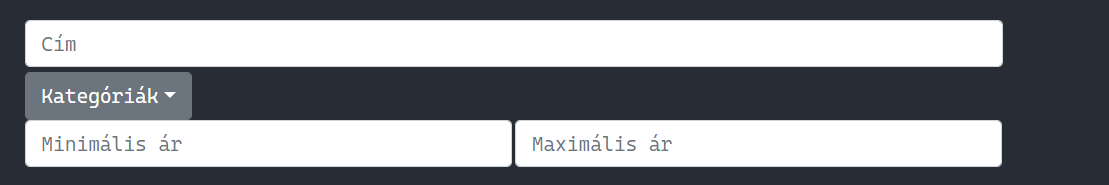
7. ábra - Jelenleg online szerverek listája



8. ábra - Szerver megtekintése cím

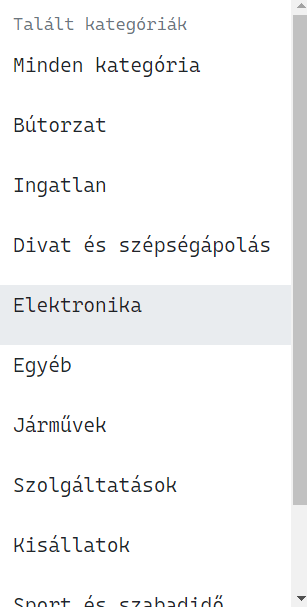
* + 1. Hirdetések megtekintése

A második menüelem az aggregátor szerverről elérhető összes publikus hirdetést hivatott egy átlátható, kereshető formában bemutatni. Említésre méltó a cím alatt elhelyezett kereső mező, amelyek lehetővé teszik a mutatott hirdetések szűrését több paraméter után: Cím, Kategória, Minimális ár, Maximális ár. A bevitt adatokkal szűrést az előzőhöz[[3]](#footnote-4) hasonló keresés gomb kattintásával lehet kezdeményezni



9. ábra - Kereső csoport

A címekre egy egyszerű, speciális karaktereket nem tartalmazó, szövegként kereshetünk rá. A hirdetés árára való megkötés a matematikai értelemben vett minimum és maximumként működik, tehát zárt halmazt alkotnak, melynek tartalmaznia kell az árat. Ezekkel ellentétben a *Kategória* filter nem egy szabadon választható, tetszőleges megkötés, hanem az összes elérhető szerver által nyilvántartott kategóriákból lehet csupán választani, egy legördülő menü formájában. Itt fontos megjegyzés, hogy bár egy Kategóriát tárol adott szerver, nem azt jelenti föltétlen, hogy jelenleg nyilvánosak abban a kategóriában hirdetések!

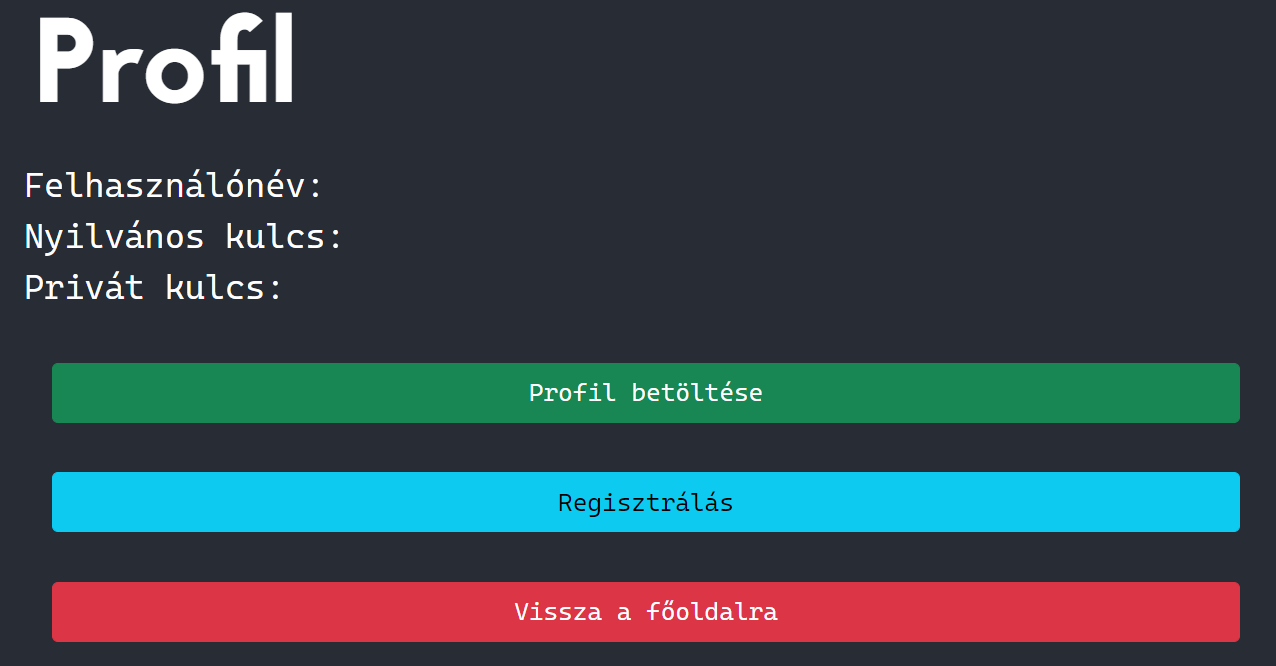


10. ábra - Kategória kiválasztó legördülő menü

[TODO]

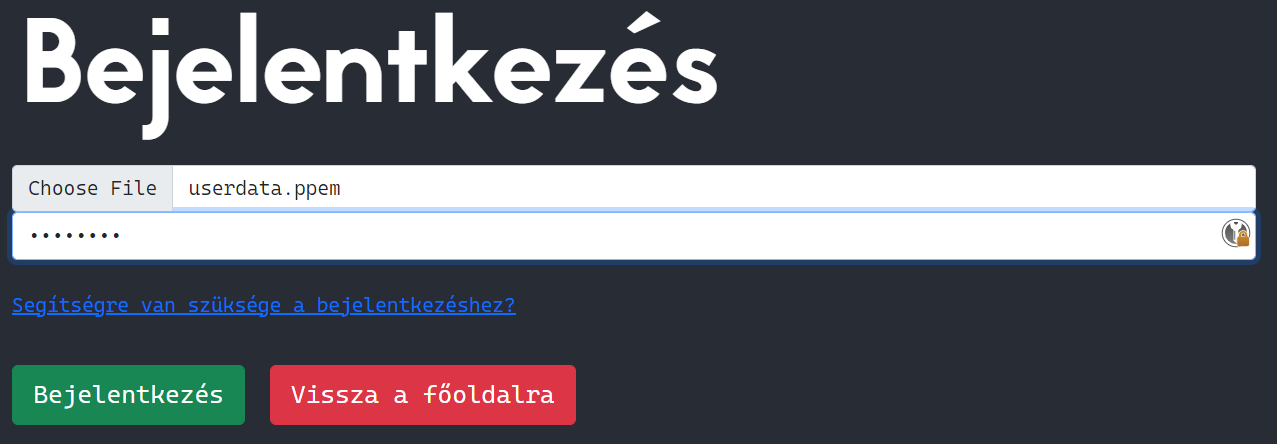
* + 1. Profil kezelése

A harmadik menüelem az a saját profilunk megtekintésére szolgál és további két menüpontra oszlik szét. Nem bejelentkezett felhasználó esetében a további választható opciók, a regisztrálás vagy a bejelentkezés.



11. ábra - Kijelentkezett profilfelület

A bejelentkezés a regisztráció során létrehozott, különleges fájl beolvasását feltételezi.



12. ábra - Bejelentkező oldal

Bejelentkezett felhasználó esetében a kijelentkezés illetve saját hirdetéseinek kezelése közül választhat. A *Kijelentkezés* feliratú gomb kiüríti a memóriából a felhasználó magánadatait, ezáltal visszatérítve őt az előző állapothoz.

Az első menüpont, azonban, átirányít egy másik oldalra, ahol a felhasználó létrehozhat egy új hirdetést vagy a már megismert módon[[4]](#footnote-5) listázva tekintheti meg az összes már általa létrehozott és épp elérhető hirdetést.



13. ábra - Sikeresen bejelentkezett felhasználó profilpanelje

[TODO]

* + 1. Még több információ a P2P-PP projektről

Végül, de nem utolsó sorban találjuk a *Több infó a projektről* feliratú gombot, mely az infó oldalra irányít át. Itt szöveges utasítások és hivatkozások segítik a felhasználó kiigazodását az oldalon és az esetleg dev felhasználóvá válási folyamatban. Az oldal alján külön színnel feltűntetve adatbázis létrehozásához szükséges lépések



14. ábra - Információs panel

1. Fejlesztői dokumentáció
   1. Követelményspecifikáció

*Alkalmazás neve:*

Peer-to-Peer Piac Portál

*Alkalmazás rövid leírása:*

A Peer-to-Peer Piac Portál egy olyan apróhirdetés platformot hivatott megvalósítani, mely egy decentralizált rendszeren alapszik. Kinézet és használat szempontjából is az egyszerűség jellemző rá.

*Funkcionális követelmények:*

* Felhasználók regisztrálása
* Felhasználók ki- és beléptetése
* Hirdetések listázása
* Hirdetés tároló szerverek listázása
* Hirdetések szűrése különböző paraméterek alapján
* Hirdetés feltöltése felhasználó által
* Listázási státusz állíthatósága (Nyilvános hirdetéseket bárki megtekinthet, míg nem nyilvános hirdetéseket csak a létrehozójuk hozhat létre)
* Aggregátor által számontartott, egymással egyenrangú szervereken történő adattárolás

Nem *funkcionális követelmények:*

* *Platform független*
* *Letisztult felhasználói felület*

*Felhasználói követelmények:*

* Egyszerű kezelhetőség
* Ne legyen szükség sütikre és azonosító magánadatok tárolására

*Biztonsági megkötések:*

* Felhasználói jelszavaknak titkosított tárolása
* HTTPS protokoll könnyű alkalmazhatósága
* Valid és biztonságos kriptográfiai eljárások használata
  1. Környezet leírása

A program fejlesztése az alábbi paraméterekkel rendelkező számítógépen történt:

* Intel Core i7-8565U @1.80GHz processzor
* 15.8GB (16.0 GB) DDR4 3200 memória
* 64-bites rendszer, x64-alapú CPU
* Windows 11 Pro 22H2 operációs rendszer

A fejlesztés túlnyomó része Visual Studio Code kiterjesztésekkel bővített szövegszerkesztőben történt, míg teszteléshez a Chromium 100 böngészőplatform beépített eszközeit, illetve a Postman nevű programot használtam.

Az alkalmazás elkészítésekor kiemelt fontosságú szempont volt a modern webapplikációkra jellemző architektúra használata, vagyis, hogy egyértelműen, logikailag és fizikailag is elkülönüljön a kliens-, a szerver- és az adatbázisréteg. Szakdolgozatom esetében a kliensoldali statikus fájlok kiszolgálását egy webszerver végzi, mely HTTP kéréseken keresztül kommunikál az aggregátor szerveroldali REST API-n keresztül az adatbázisokkal.

* 1. Felhasznált technológiák
     1. Architektúra

1. Tesztelés
2. Továbbfejlesztési lehetőségek
   1. Kriptobiztonsági fejlesztési javaslatok

Bár a használt RSA protokoll (RSA-Probabilistic Signature Scheme) megfelel a jelen hétköznapi kihívásainak, hiszen a PKCS#1 ver. 1.5 leírásai szerint nincs ismert elméleti sebezhetősége, az nem azt jelenti, hogy ez a jövőben is helyt áll. Ajánlott szemmel tartani az aláíró és ellenőrző algoritmusokat.

Ugyanakkor a mellékelt implementáció nem vet be minden ismert vagy ajánlott elővigyázatot. Ennek több oka is van.

* 1. Vizuális fejlesztések
  2. Több nyelv

A programozási nyelv kiválasztásában fontos szerepet töltött be, hogy napjainkban ez a leg jövőbiztosabb környezet, azonban a technológia fejlődésével a programozási nyelvek fejlődése is várható. Tehát, egy nyílt forráskódú projekt fenntarthatóságát csak javítaná annak több nyelven való, megegyező működést biztosító implementálása. Véleményem szerint egy érdemes kiindulópont a két szervert IIS Express segítségével C#-ban is létrehozni ASP.NET keretrendszerben.

* 1. Funkcionális javítások

1. Összefoglalás

Azért választottam szakdolgozatom témájának a BitTorrent szerű Peer-to-Peer rendszer kialakítását, mert szerettem volna egy napjainkban aktuális és kiemelkedő fontosságú változásnak aktív szereplője lenni; és reméltem, hogy munkám eredménye valóban hasznos és biztonságos irányba segít elmozdítani a *status-quo*t.

Az alkalmazás megtervezése és fejlesztése során mindvégig szem előtt tartottam, hogy az könnyen érthető és kezelhető legyen, bárki számára elérhető maradjon. Szeretném, hogy ilyen és hasonló megvalósítások segítsenek az emberek hétköznapjaiba vinni a peer-to-peer architektúrákat, hiszen a modern kor kirobbanó internetfogyasztása komoly terhet rak az öregedni mutatkozó centralizált internet infrastruktúra hátára. Igyekeztem továbbá a kódot úgy megírni, hogy az könnyedén kibővíthető legyen az esetleges jövőbeli továbbfejlesztésekor.

Nem utolsó sorban, kimondottan vonzott a kihívás, amit az új technológiák jelentenek, hiszen, bár az elméleti háttérben szükségszerűen jártas voltam, mint kriptobiztonság érdeklődő, a gyakorlati megvalósítás elemei közül valamennyi többnyire ismeretlen volt számomra. Így tehát, felbecsülhetetlenül értékes tapasztalatot szereztem a TypeScript/JavaScript nyelvben, az Express keretrendszerben, a React frontend keretrendszerben, és a webfejlesztéshez kapcsolódó kriptobiztonságban általánosan. Nagyon örülök, hogy ezekkel a technológiákkal dolgozhattam, így is bővítve szakmai tudásomat.

1. Köszönetnyilvánítás

Szeretném ezúton is megköszönni témavezetőmnek és konzulensemnek, Kovács Attilának, véget nem érő türelmét, tervezői meglátását, szüntelen új perspektívából felvilágosító tanácsait, korlátlan elméleti mentorálását és tapasztalatot bizonyító meglátásait. Ugyanakkor a gyakorlati implementáció lényegében lehetetlen lett volna pszichológusom, Csizovszky Ferenc, pszichiáterem, dr. Nagy Attila István, illetve szerető, végtelenül támogató és megértő családom és barátaim segítsége nélkül, főleg Andi Péter felfoghatatlan mennyiségű morális és technikai útmutatása nélkül. Legyen e szakdolgozat milliomodik példája annak, hogy a helyénvaló segítséggel bármilyen munka megvalósítható!

1. Irodalomjegyzék
2. W3Schools HTML: <https://www.w3schools.com/html/default.asp> (2022. 03. 26.)
3. W3Schools CSS: [https://www.w3schools.com/css/default.asp](https://www.w3schools.com/css/default.asp%20)  
   (2022. 03. 26.)
4. TypeScript: <https://www.typescriptlang.org> (2022. 03. 26.)
5. Node.js: <https://nodejs.org/en/docs/> (2022. 04. 02.)
6. Node Package Manager: <https://www.npmjs.com> (2022. 04. 02.)
7. ExpressJS: <https://expressjs.com> (2022. 04. 05.)
8. SQLite: <https://sqlite.org/docs.html> (2022. 04. 11.)
9. Prisma ORM: <https://www.prisma.io> (2022. 04. 11.)
10. ReactJS: <https://reactjs.org> (2022. 04. 28.)
11. Reactstrap: <https://reactstrap.github.io/> (2022. 05. 01.)
12. SASS: <https://sass-lang.com> (2022. 05. 02.)
13. Rose, Scott: [Planning for a Zero Trust Architecture: A Planning Guide for Federal Administrators](https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.20.pdf) (2022. 05. 06.)
14. Web Crypto API: <https://developer.mozilla.org/en-US/docs/Web/API/Web_Crypto_API> (2022. 04. 16.)
15. Bellare, Mihir; Rogaway, Philip: PSS: [Provably Secure Encoding Method for Digital Signatures](https://web.archive.org/web/20040713140300/http:/grouper.ieee.org/groups/1363/P1363a/contributions/pss-submission.pdf) (1998. 08.)
16. Mellékletek

* 1db USB pendrive, melynek tartalma
  + Az alkalmazás forráskódja
  + E dolgozat pdf formátumban
  + A teszteléshez használt fájlok

1. Lásd: Az adatbázis telepítése, létrehozása és futtatása [↑](#footnote-ref-2)
2. Lásd: Profil kezelése [↑](#footnote-ref-3)
3. Lásd: Elérhető szerverek megtekintése [↑](#footnote-ref-4)
4. Lásd: Hirdetések megtekintése [↑](#footnote-ref-5)