Budapesti Műszaki Szakképzési Centrum

Verebély László Technikum

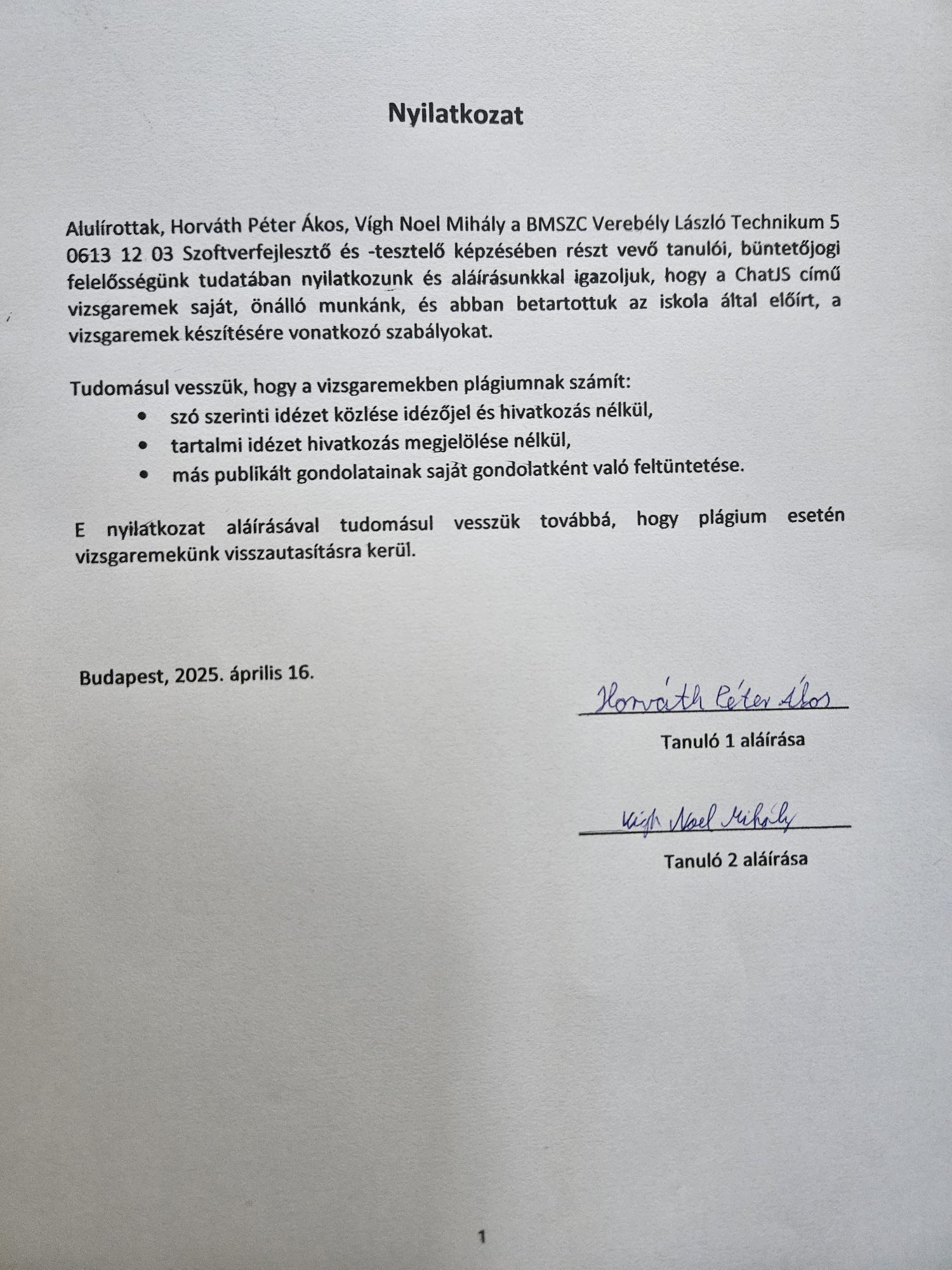
5-0613-12-03 Szoftverfejlesztő és tesztelő

ChatJS

Készítették: Horváth Péter Ákos, Vígh Noel Mihály

Konzulens tanárok: Somogyi Erika, Horváth Attila

Budapest, 2025. április



# Tartalomjegyzék

[Tartalomjegyzék 3](#_Toc195698163)

[1. Bevezetés 4](#_Toc195698164)

[2. Fejlesztői dokumentáció 5](#_Toc195698165)

[2.1. Alkalmazott fejlesztő eszközök 5](#_Toc195698166)

[2.2. Alkalmazott technológiák 5](#_Toc195698167)

[2.3. Futtatási környezet 5](#_Toc195698168)

[2.4. Fejlesztési ütemterv 5](#_Toc195698169)

[2.5. Fejlesztés menete 5](#_Toc195698170)

[2.5.1. Piackutatás 5](#_Toc195698171)

[2.5.2. Adatbázis megtervezése 6](#_Toc195698172)

[2.5.3. Asztali alkalmazás főbb algoritmusai 14](#_Toc195698173)

[2.5.4. Frontend alkalmazás főbb algoritmusai // Webalkalmazás főbb algoritmusai 21](#_Toc195698174)

[2.5.5. Backend alkalmazás főbb algoritmusai 27](#_Toc195698175)

[2.6. Tesztelés 32](#_Toc195698176)

[3. Felhasználói dokumentáció 33](#_Toc195698177)

[3.1. ChatJS asztali alkalmazás 33](#_Toc195698178)

[3.2. ChatJS webalkalmazás 38](#_Toc195698179)

[4. Továbbfejlesztési lehetőségek 51](#_Toc195698180)

[5. Összegzés 52](#_Toc195698181)

[Ábrajegyzék 53](#_Toc195698182)

# Bevezetés

A vizsgaremekünk témáját személyes tapasztalataink ihlették: egy online chatalkalmazás fejlesztését választottuk, mivel mindig is érdekelt bennünket, hogyan működnek a valós idejű kommunikációs platformok. Napjainkban, amikor a távolságtartás és a digitális kapcsolattartás egyre nagyobb jelentőséget kap, úgy gondoltuk, hogy egy egyszerű és hatékony chatalkalmazás hasznos eszköz lehet a mindennapi kommunikációban.

Az ötlet abból született, hogy mindketten gyakran használunk különféle üzenetküldő szolgáltatásokat, és szerettük volna megérteni, hogy miként működnek a háttérben.

Célunk egy olyan felhasználóbarát és gyorsan működő chatalkalmazás létrehozása volt, amely nemcsak technikai ismereteinket gazdagítja, hanem mások számára is hasznos eszköz lehet. A projekt során különös figyelmet fordítottunk a valós idejű üzenetküldés megvalósítására, a felhasználói felület egyszerűsítésére, valamint a rendszer megbízhatóságára. A közös munka eredményeként egy modern kommunikációs platformot sikerült megvalósítanunk.

# Fejlesztői dokumentáció

## Alkalmazott fejlesztő eszközök

* Kódszerkesztő: Visual Studio, Visual Studio Code
* Fejlesztői környezet: XAMPP

## Alkalmazott technológiák

* Programozási nyelvek: JavaScript, C#
* Webes technológiák: HTML, CSS
* Adatbázis motor: MariaDB (MySQL nyelvvel)

## Futtatási környezet

Az oldal kompatibilis minden korszerű böngészővel, és a reszponzív kialakításának köszönhetően kiválóan használható mobil- és táblagépeken. Emellett elérhető egy Windows operációs rendszerre optimalizált asztali alkalmazás is, amely az egyszerű és kényelmes használatot támogatja.

## Fejlesztési ütemterv

|  |  |
| --- | --- |
| Határidő | Feladat |
| 2024. december 19 | Témaválasztás, csoportalakítás |
| 2025. január 27 | Dizájnterv elkészítése: weboldal sablon, Figma |
| 2025. február 21 | Adatbázisterv elkészítése |
| 2025. február 24 | Autentikáció elkészítése |
| 2025. március 31 | A weboldal és az asztali alkalmazás főbb funkcióinak kialakítása |
| 2025. április 7.-11 | Vizsgaremek előzetes értékelése, hibajavítás |
| 2025. április 16 | Vizsgaremek leadása |

## Fejlesztés menete

### Piackutatás

A fejlesztés megkezdése előtt részletesen elemeztünk több népszerű csevegő alkalmazást. Elemzésünk során különös figyelmet fordítottunk az egyszerű kezelőfelületre és a valós idejű kommunikációra, hiszen ezek váltak a saját alkalmazásunk alapvető jellemzőivé. A meglévő megoldások nemcsak igazolták az ötletünk életképes voltát, hanem értékes inspirációt is nyújtottak.

### Adatbázis megtervezése

A weboldal hátterében egy MySQL adatbázis fut MariaDB motorral. Az adatbázis 11 táblából áll. A táblákat logikusan és következetesen vettük fel, illetve igyekeztünk a redundanciát elkerülni. A legnagyobb hangsúlyt a felhasználók és az üzenetek kapták. A továbbiakban leírjuk a táblák felépítését:

attachment

|  |  |
| --- | --- |
| attachmentId | egész, a melléklet azonosítója, automatikus sorszámozás, PK |
| fileType | szöveges, a melléklet kiterjesztése, kötelező kitöltésű |
| filePath | szöveges, a melléklet elérési útvonala, kötelező kitöltésű |
| fileSize | lebegő pontos, a melléklet mérete, kötelező kitöltésű |

blocked

|  |  |
| --- | --- |
| blockedId | egész, a tábla azonosítója, automatikus sorszámozás, PK |
| userId | egész, a tiltást küldő felhasználó azonosítója, kötelező kitöltésű, FK |
| blockedUserId | egész, a tiltott felhasználó azonosítója, kötelező kitöltésű, FK |
| blockedAt | dátum, a tiltás küldésének dátuma, alapértelmezett aktuális dátum |

bugreports

|  |  |
| --- | --- |
| bugReportId | egész, a hiba azonosítója, automatikus sorszámozás, PK |
| senderUserid | egész, a hiba küldőjének az azonosítója, kötelező kitöltésű, FK |
| header | szöveges, a hiba címe, kötelező kitöltésű |
| description | szöveges, a hiba részletes leírása, kötelező kitöltésű |
| attachmentId | egész, a hibárol készült kép azonosítója, alapértelmezett NULL, FK |
| sentAt | dátum, a hiba küldésének dátuma, alapértelmezett aktuális dátum |
| isClosed | rövid egész, a hibajegy lezárt-e, 0 vagy 1 az értéke, alapértelmezett 0 |

conversations

|  |  |
| --- | --- |
| conversationId | egész, a csevegés azonosítója, automatikus sorszámozás, PK |
| conversationName | szöveges, a csevegés neve, kötelező kitöltésű |
| createdAt | dátum, a csevegés létrehozásának dátuma, alapértelmezett aktuális dátum |
| isGroupChat | rövid egész, a csevegés csoportos beszélgetés-e, 0 vagy 1 az értéke, alapértelmezett 0 |

conversationusers

|  |  |
| --- | --- |
| conversationUserId | egész, a tábla azonosítója, automatikus sorszámozás, PK |
| userId | egész, a csevegésben lévő felhasználó azonosítója, kötelező kitöltésű, FK |
| conversationId | egész, a csevegés tábla azonosítója, kötelező kitöltésű, FK |
| isAdmin | rövid egész, a csevegés adminja-e a felhasználó, 0 vagy 1 az értéke, alapértelmezett 0 |
| joinedAt | dátum, a csevegésbe való belépés dátuma, alapértelmezett aktuális dátum |

friendrequest

|  |  |
| --- | --- |
| requestId | egész, a kérés azonosítója, automatikus sorszámozás, PK |
| senderUserId | egész, a küldő felhasználó azonosítója, kötelező kitöltésű, FK |
| receiverUserId | egész, a vevő felhasználó azonosítója, kötelező kitöltésű, FK |
| sentAt | dátum, a kérelem küldésének dátuma, alapértelmezett aktuális dátum |
| isTimedOut | rövid egész, a kérelem érvényes-e még, 0 vagy 1 az értéke, alapértelmezett 0 |
| status | enum, az kérés státusza, értékei: “pending”; “accepted”; “rejected”, alapértelmezett “pending” |

friends

|  |  |
| --- | --- |
| friendId | egész, a tábla azonosítója, automatikus sorszámozás, PK |
| userId | egész, a felhasználó azonosítója, kötelező kitöltésű, FK |
| friendUserId | egész, a barát felhasználó azonosítója, kötelező kitöltésű, FK |
| friendedAt | dátum, a barát hozzáadásának ideje, alapértelmezett aktuális dátum |

logins

|  |  |
| --- | --- |
| loginId | egész, az bejelentkezés azonosítója, automatikus sorszámozás, PK |
| userId | egész, a felhasználó azonosítója, kötelező kitöltésű, FK |
| loginDate | dátum, a bejelentkezés dátuma, alapértelmezett aktuális dátum |
| logoutDate | dátum, a kijelentkezés ideje, alapértelmezett aktuális dátum |
| isLoggedIn | rövid egész, be van e jelentkezve a felhasználó, 0 vagy 1 az értéke, alapértelmezett 1 |
| location | szöveges, a bejelentkezés helye, kötelező kitöltésű |
| ipAddress | szöveges, a bejelentkezés IP címe, kötelező kitöltésű |
| deviceType | szöveges, az bejelentkezett eszköz típusa, kötelező kitöltésű |
| deviceName | szöveges, az bejelentkezett eszköz neve, kötelező kitöltésű |
| userAgent | szöveges, a böngésző megnevezése, kötelező kitöltésű |

messages

|  |  |
| --- | --- |
| messageId | egész, az üzenet azonosítója, automatikus sorszámozás, PK |
| conversationid | egész, a beszélgetés azonosítója, kötelező kitöltésű, FK |
| senderUserId | egész, a küldő felhasználó azonosítója, kötelező kitöltésű, FK |
| sentAt | időbélyeg, az üzenet küldésének pontos ideje, alapértelmezett aktuális időpont |
| state | enum, az üzenet státusza, értékei: “sent”; “delivered”; “seen”, alapértelmezett “sent” |
| isDeleted | rövid egész,törölt-e az üzenet, 0 vagy 1 az értéke, alapértelmezett 0 |
| isEdited | rövid egész, szerkesztett-e az üzenet, 0 vagy 1 az értéke, alapértelmezett 0 |
| messageText | szöveges, alapértelmezett NULL |
| attachmentId | egész, csatolt melléklet azonosítója, alapértelmezett NULL |
| replyTo | egész, válaszolt üzenet azonosítója, alapértelmezett NULL |
| isReported | rövid egész,jelentett-e az üzenet, 0 vagy 1 az értéke, alapértelmezett 0 |

themes

|  |  |
| --- | --- |
| themeId | egész, a témak azonosítója, automatikus sorszámozás, PK |
| themeName | szöveges, a téma megnevezése, kötelező kitöltésű |

users

|  |  |
| --- | --- |
| userId | egész, a felhasználó azonosítója, automatikus sorszámozás, PK |
| displayName | szöveges, a felhasználó megjelenítési neve, 20 karakter hosszú, kötelező kitöltésű |
| displayId | egész, a felhasználó látható azonosítója, 4 karakter hosszú, kötelező kitöltésű |
| email | szöveges, a felhasználó email címe, egyedi, kötelező kitöltésű |
| telephone | szöveges, a felhasználó telefonszáma, egyedi |
| password | szöveges, a felhasználó jelszava, kötelező kitöltésű |
| createdAt | dátum, a fiók létrehozásának dátuma, kötelező kitöltésű |
| updatedAt | dátum, a fiók legutolsó frissítésének időpontja, |
| isOnline | rövid egész, elérhető-e a felhasználó, 0 vagy 1 az értéke, alapértelmezett 0 |
| isLookingForFriends | rövid egész, keres-e barátokat a felhasználó, 0 vagy 1 az értéke, alapértelmezett 0 |
| profilePicPath | szöveges, profilkép elérési útvonala, alapértelmezett NULL |
| isSiteAdmin | rövid egész, admin-e a felhsználó, 0 vagy 1 az értéke, alapértelmezett 0 |
| currentThemeid | egész, a jelenleg kiválasztott oldal témája, alapértelmezett 1, FK |
| bio | szöveges, a felhasználóról rövid leírás, alapértelmezett NULL |
| isDeleted | rövid egész, törölt-e a felhasználó, 0 vagy 1 az értéke, alapértelmezett 0 |
| isBanned | rövid egész, kitiltott-e a felhasználó, 0 vagy 1 az értéke, alapértelmezett 0 |

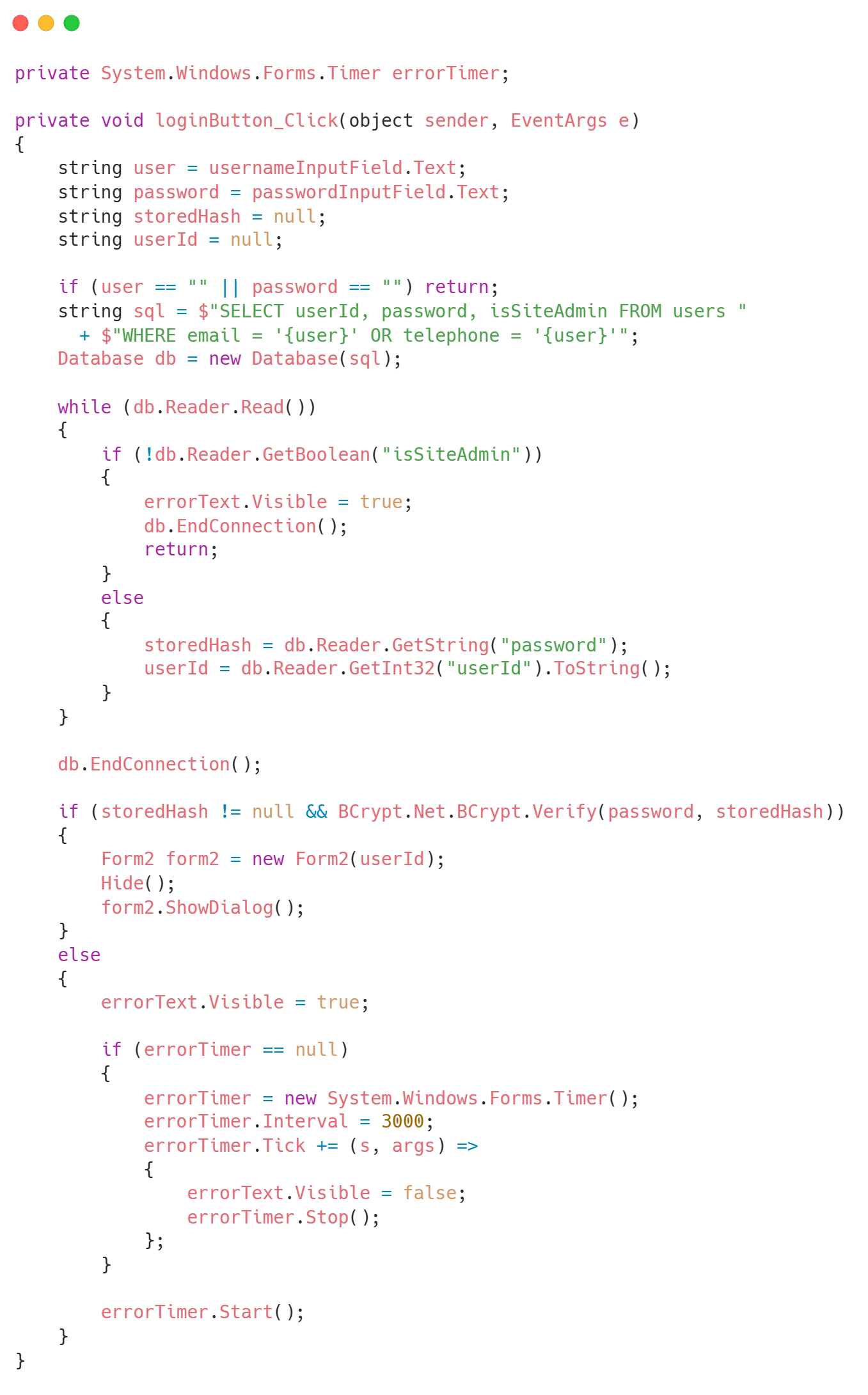
Az elsődleges kulcsokat PK, az idegen kulcsokat FK jelöléssel láttuk el.



1. ábra EER ábra a chatdb adatbázisról

### Asztali alkalmazás főbb algoritmusai

Az asztali felületre csak az oldalt kezelő felhasználók tudnak bejelentkezni.



. ábra Bejelentkezés függvénye

A loginButton\_Click eseménykezelő célja, hogy a felhasználó által megadott adatok alapján hitelesítse a bejelentkezést, és sikeres hitelesítés esetén továbbléptesse a felhasználót az admin felületre.

A függvény a felhasználó által beírt e-mail címet vagy telefonszámot, illetve a jelszót olvassa ki a megfelelő input mezőkből.

Adatbázis-lekérdezés:

* A rendszer SQL-lekérdezést hajt végre a users táblán, ahol az e-mail vagy telefonszám egyezik a megadott értékkel.
* Az adatok beolvasása során ellenőrzi, hogy az adott felhasználó rendelkezik-e admin jogosultsággal (isSiteAdmin mező).
* Amennyiben nem admin, a hibajelzés (errorText) megjelenik, az adatbázis kapcsolat lezárul, és a folyamat megszakad.
* Amennyiben admin, a jelszó hash (password) és a felhasználói azonosító (userId) kinyerésre kerül.

Jelszó ellenőrzése és navigáció:

* A lekért jelszó-hash és a felhasználó által megadott jelszó összehasonlítása a BCrypt könyvtár segítségével történik.
* Sikeres hitelesítés esetén a jelenlegi űrlap (Form1) elrejtésre kerül, és megnyílik a második űrlap (Form2), amely az admin felületet reprezentálja.

Hibakezelés:

* Sikertelen hitelesítés esetén a errorText komponens láthatóvá válik, ami 3 másodperc után autómatikusan eltűnik.

Az asztali felületre való belépés után a felhasználó többféle dolgot is kezelhet. Ezek közül az egyik a hibajelentések kezelése.



. ábra A jelentések betöltése algoritmus

A LoadBugs metódus célja, hogy a hibajelentéseket megjelenítse egy táblázatos nézetben (reportsDataGrid), a isClosed paraméter értékének megfelelően (lezárt vagy nyitott jelentések szerint szűrve).

Lekérdezés összeállítása és végrehajtása:

* A isClosed logikai érték alapján egy numerikus érték (1 vagy 0) kerül meghatározásra a lekérdezéshez.
* Az SQL-lekérdezés a users és bugreports táblákból olvassa ki a szükséges adatokat INNER JOIN segítségével, csak a megadott státuszú hibajelentésekre szűrve.
* Az adatbázis olvasása során minden találathoz kiszedi a bejelentő nevét, azonosítóját, a hibajelentés fejlécét és az egyedi hibajelentés-azonosítót.

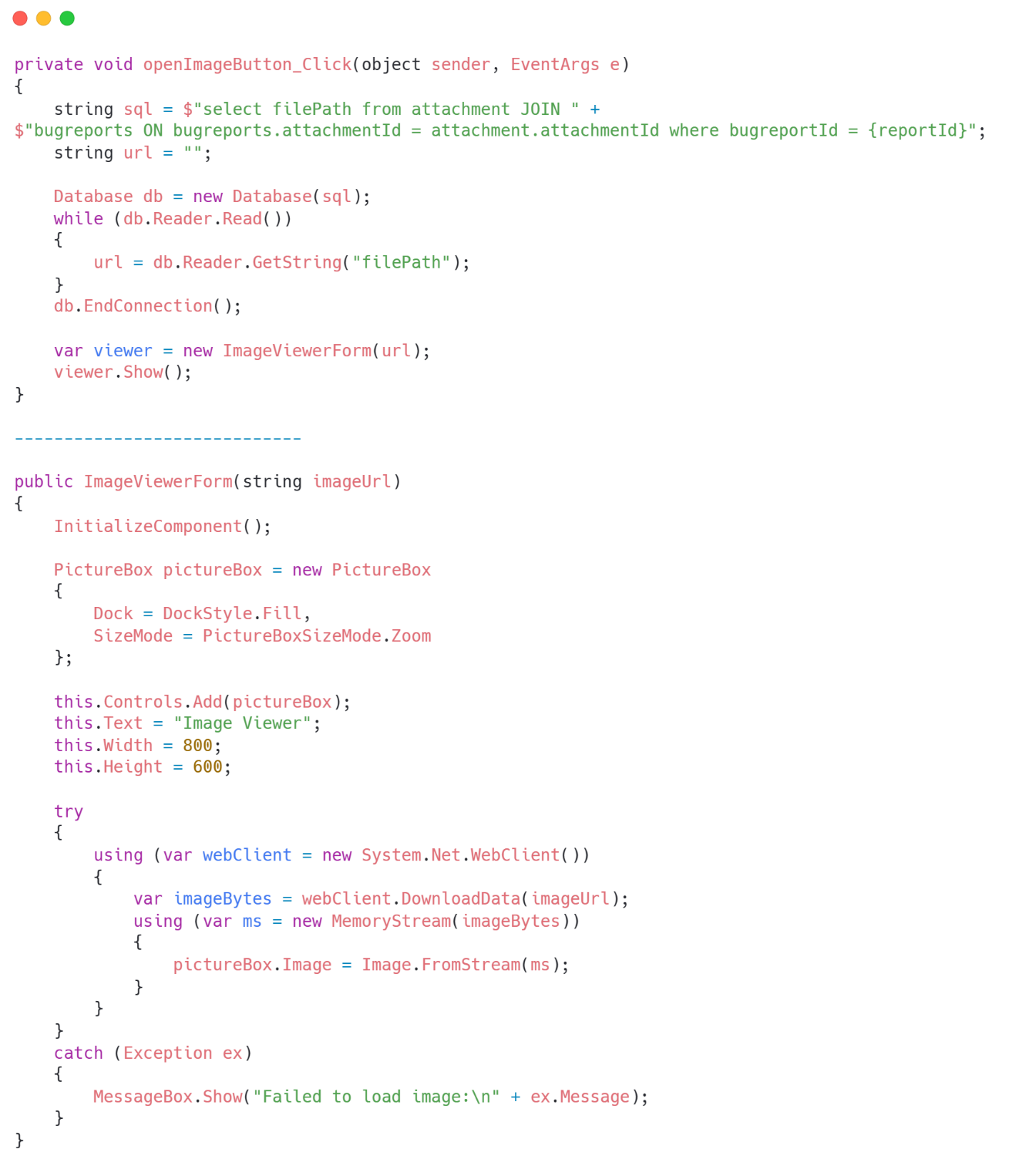
Adatok megjelenítése:

* Az egyes hibajelentések adatai új sorokként kerülnek hozzáadásra a reportsDataGrid komponenshez.

Felület frissítése:

* A hibajelentés részleteit megjelenítő komponensek (titleText, doneButton, openImageButton) elrejtésre kerülnek.
* A leírás szövegmezeje (descriptionText) kiürül.
* Az előzőleg kiválasztott sor referenciaértéke (prewRow) nullára állítódik, jelezve, hogy jelenleg nincs kiválasztva jelentés.

A hiba elküldésekor képet is lehet feltölteni, amelyet az asztali felületen meg lehet nézni.



. ábra Kép megjelenítése algoritmusok

A ImageViewerForm osztály konstruktora egy kép URL alapján jelenít meg egy képet külön ablakban, reszponzív megjelenéssel.

Felület inicializálása:

* A InitializeComponent metódus meghívása után létrejön egy PictureBox komponens, amely a form teljes felületét lefedi (Dock = Fill), és a képarányt megtartva méretezi a képet (SizeMode = Zoom).
* A PictureBox komponens hozzáadásra kerül az űrlap vezérlőihez.
* Az űrlap címe ("Image Viewer") és alapméretei (800×600 pixel) szintén beállításra kerülnek.

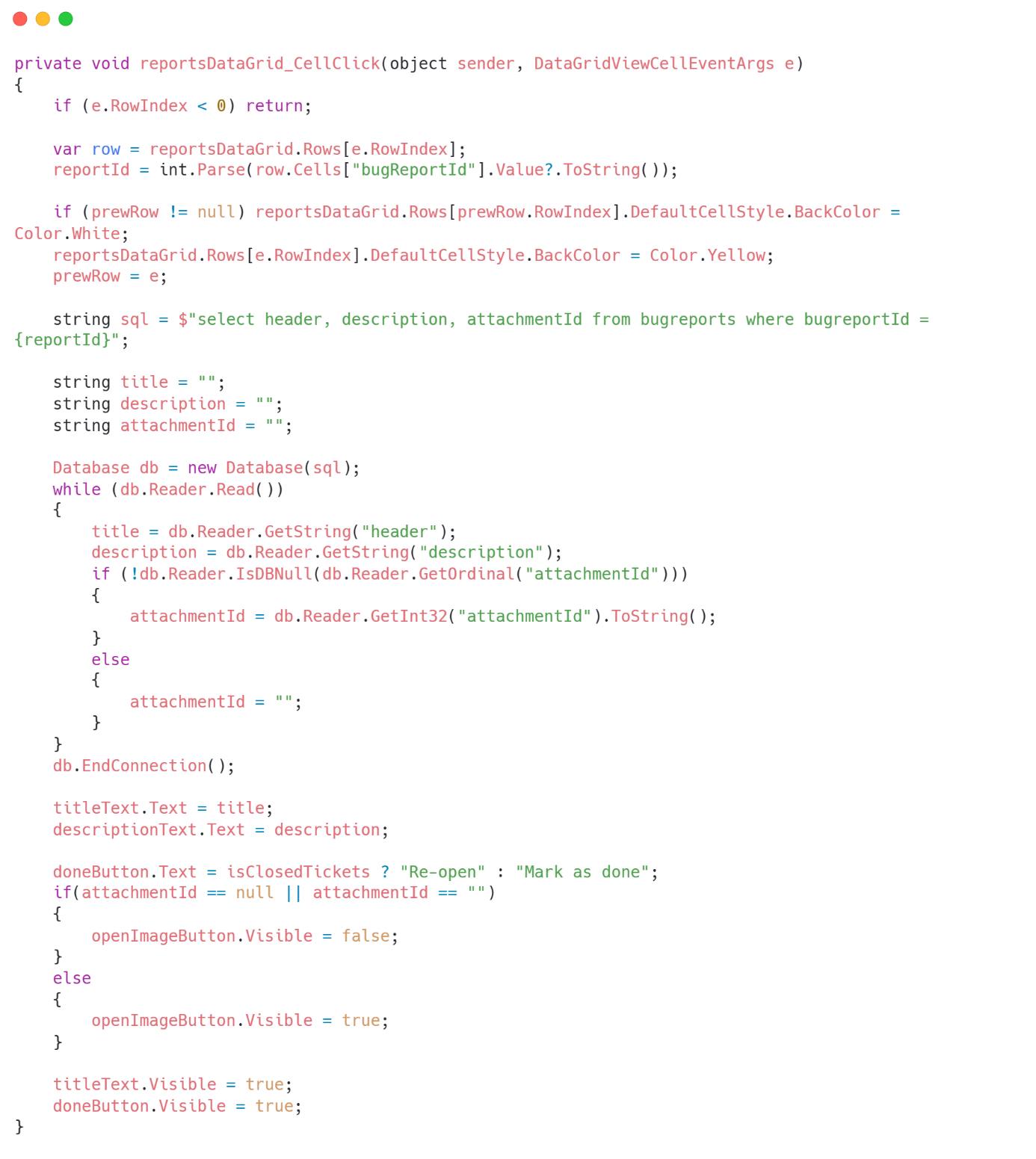
Kép betöltése URL alapján:

* A konstruktor megkísérli letölteni a képet a megadott URL-ről a System.Net.WebClient osztály segítségével.
* A letöltött bájtokat MemoryStream objektumba tölti, amelyből a PictureBox.Image tulajdonság feltöltése történik (Image.FromStream).

Hibakezelés:

* Amennyiben a kép letöltése vagy feldolgozása során hiba történik (pl. érvénytelen URL, hálózati probléma), a kivétel lekezelésre kerül, és a felhasználó hibaüzenetet kap.

Az összes jelentés egy táblában jelenik meg. A cellákra kattintva egy szövegdobozban jelenik meg a leírás, alatta pedig gombok a kezeléshez.



. ábra Cell click algoritmus

Sor kiválasztása és háttérszín kezelése:

* A kiválasztott sorból kiolvassa a jelentett üzenet azonosítóját (reportedMessageId), és elmenti azt a messageId változóba.
* Amennyiben egy másik sor már korábban ki volt jelölve, annak háttérszíne visszaáll fehérre.
* Az aktuálisan kiválasztott sor háttérszíne sárgára változik, vizuálisan jelezve az aktív kijelölést.

Üzenet részleteinek lekérdezése:

* SQL-lekérdezéssel kapcsolja össze a messages és users táblákat, és lekéri a kiválasztott üzenet szövegét (messageText), küldési idejét (sentAt), törölt státuszát (isDeleted), illetve a felhasználó tiltásának állapotát (isBanned).
* Az üzenet küldési ideje formázva jelenik meg (pl. 2024. 04. 16.).

Felhasználói felület frissítése:

* Az üzenet szövege és dátuma megjelenik a megfelelő mezőkben (messageText, dateText).
* A deleteMessageButton és a banUserButton láthatósága az adott üzenet és felhasználó állapotához igazodik:
* Ha az üzenet már törölve van, a törlés gomb rejtve marad.
* Ha a felhasználó már tiltva van, a tiltás gomb szintén nem jelenik meg.
* A closeReportButton gomb minden esetben láthatóvá válik, lehetőséget adva a jelentés lezárására.

### Frontend alkalmazás főbb algoritmusai // Webalkalmazás főbb algoritmusai

A rendszer támogatja a korábban elküldött üzenetek törlését, amely valós időben frissíti a felhasználói felületet, és szinkronban áll a backend adattárolásával.



. ábra Üzenet törlése algoritmus

A deleteMessage függvény célja, hogy egy megadott üzenetet töröljön a felhasználó felületéről, és az erről szóló információt továbbítsa a szerver felé.

Helyi állapot frissítése:

* A setMessages állapotkezelő függvény segítségével a kliens frissíti a helyi üzenetlistát, és a törölt üzenethez isDeleted: 1 értéket rendel.
* Emellett a setFriends függvény módosítja a „lastMessage” értéket az adott beszélgetésnél, ezzel jelezve, hogy az utolsó üzenet törölve lett.

A rendszer lehetőséget biztosít arra, hogy a felhasználók barátokat adhassanak hozzá egymáshoz egyedi azonosító és felhasználónév alapján. A barátfelvétel folyamata egyszvalmeg HTTP POST kérésen és Socket.IO eseményen keresztül, így a címzett felhasználó valós időben értesítést kaphat az új barátkérelemről.



. ábra Barát hozzáadás algoritmus

A handleAddFriend függvény a barátkérés küldéséért felelős művelet, amely a következő lépésekből áll:

Adatellenőrzés: A függvény első lépésként ellenőrzi, hogy a felhasználó kitöltötte-e a szükséges mezőket (receiverUserName, receiverDisplayId). Amennyiben nem, a művelet megszakad.

Állapotkezelés: A folyamat alatt az alkalmazás isLoading és loadingTime állapotokat aktiválja, valamint alaphelyzetbe állítja a státuszüzenetet.

HTTP POST kérés: A rendszer POST kérést küld a /api/friends/add végpontra, amely a felhasználói azonosítók alapján dolgozza fel a barátkérést:

* senderUserId: a jelenlegi felhasználó azonosítója
* receiverUserName: a címzett felhasználóneve
* receiverDisplayId: a címzett publikus azonosítója

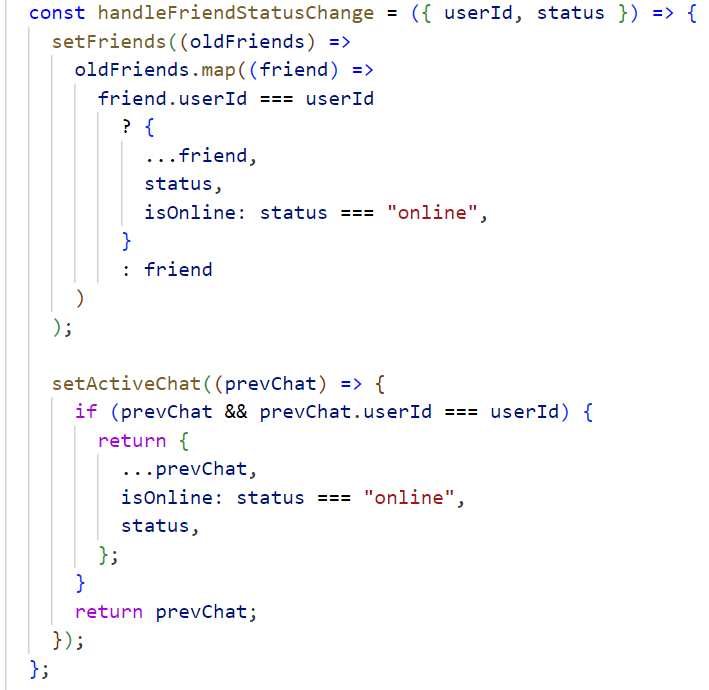
Socket.IO esemény: A kérés sikeres feldolgozása után a kliens friend\_request Socket.IO eseményt küld, amely értesíti a címzettet valós időben, ha az online állapotban van.

Visszajelzés és felhasználói élmény:

* Siker esetén a felület megjeleníti a sikeres kérelem státuszát, valamint automatikusan törli a bevitt adatokat.
* Hiba esetén egy hibaüzenet jelenik meg.
* Az állapotüzenetek 5 másodperc után automatikusan törlésre kerülnek.

A modern csevegőalkalmazások egyik alapvető követelménye a felhasználók valós idejű online státuszának kezelése és megjelenítése. A handleFriendStatusChange nevű kliensoldali

függvény felelős a felhasználói státuszváltozások (pl. online, offline) dinamikus követéséért, amely egy Socket.IO esemény hatására kerül meghívásra.



. ábra Státusz változás algoritmus

A függvény frissíti a lokális állapotban tárolt felhasználók adatait, hogy tükrözze az adott ismerős új státuszát (status), valamint az isOnline logikai értéket. Továbbá, ha az éppen aktív

chatpartner státusza változik, akkor az activeChat állapot is frissítésre kerül, biztosítva a felület azonnali reagálását.

1. Paraméterek

A függvény egy objektumot vár bemenetként:

* userId: a felhasználó egyedi azonosítója, akinek a státusza megváltozott
* status: az új státuszérték (pl. "online", "offline", stb.)

1. Ismerősök állapotának frissítése

A setFriends függvény segítségével a meglévő ismerősök listájában (friends) frissítésre kerül az adott felhasználó:

Ahol userId egyezik, ott a status mező új értéket kap.

Emellett az isOnline boolean érték is beállításra kerül attól függően, hogy a státusz "online"-e.

1. Aktív beszélgetés frissítése

Amennyiben az aktuálisan megnyitott beszélgetés (activeChat) azonos az éppen státuszt változtató felhasználóval, frissül az ő status és isOnline mezője is. Ez biztosítja, hogy a felület azonnal tükrözze a státuszváltozást, a további információk panelen is.

A felhasználói élmény szempontjából kiemelten fontos, hogy az egyes üzenetekhez tartozó időbélyegek kontextusfüggően, könnyen értelmezhető formában jelenjenek meg. A

formatMessageTime függvény ezt a célt szolgálja azzal, hogy a kapott időbélyeget (timestamp) különböző szempontok szerint formázza.



. ábra Idő formázása algoritmus

A függvény a paraméterként megadott időbélyeg (timestamp) alapján emberi olvasásra optimalizált formátumú szöveget ad vissza. A megjelenés dinamikusan igazodik az aktuális dátumhoz, és az alábbi eseteket különbözteti meg:

* Az üzenet ma érkezett
* Az üzenet tegnap érkezett
* Az üzenet az aktuális évben érkezett, de nem ma vagy tegnap
* Az üzenet korábbi évből származik

### Backend alkalmazás főbb algoritmusai

A backend alkalmazás valós idejű kommunikációját a Socket.IO könyvtár biztosítja, amely lehetővé teszi a kétirányú, eseményalapú adatátvitelt a szerver és a kliensek között. Az alábbi algoritmus célja, hogy egy csoportos beszélgetés létrehozása után értesítse az összes résztvevő felhasználót, amennyiben azok aktív kapcsolatban állnak a szerverrel.



. ábra Csoport létrehozás socket

1. create\_group\_chat esemény

Ez az esemény akkor fut le, amikor egy új csoportos beszélgetés jön létre. A szerver először belépteti az aktuális socketet a beszélgetéshez tartozó szobába (a conversationId alapján), majd végigiterál a megadott felhasználók azonosítóin (userIds), hogy az adott felhasználókhoz tartozó socket kapcsolatokat is csatlakoztassa az adott szobához.

Az algoritmus lépései:

* Az esemény data objektumából kinyerésre kerül a beszélgetés azonosítója és a résztvevő felhasználók listája.
* Minden felhasználóra lekérdezi, hogy aktív socketkapcsolattal rendelkezik-e az connectedUsers objektum alapján.
* Amennyiben a felhasználó online, és a socket példány érvényesen lekérhető, automatikusan csatlakoztatásra kerül az adott beszélgetés szobájához
* Ha a felhasználó offline, a rendszer naplózza, hogy nincs további teendő.

Ez biztosítja, hogy minden elérhető felhasználó technikailag csatlakozzon a beszélgetéshez, és ezáltal a később érkező üzeneteket automatikusan megkaphassa.

1. show\_group\_chat esemény

Miután a beszélgetés létrejött, a rendszer show\_group\_chat eseményt bocsát ki, hogy a felhasználók számára vizuálisan is megjelenítse az új csoportos beszélgetést. Az esemény tartalmazza a beszélgetés nevét, azonosítóját, a résztvevők listáját és a létrehozás időpontját.

Az esemény működése:

* A userIds tömb mentén végighaladva a szerver lekérdezi az adott felhasználó socketkapcsolatát.
* Az online felhasználók esetében a rendszer render\_group\_chat eseményt bocsát ki, amely kliensoldalon elindítja a felület frissítését, így a beszélgetés megjelenik a felhasználó számára.
* Az offline állapotú felhasználók esetén csak naplózás történik.

Ez az esemény garantálja, hogy a beszélgetéshez tartozó információk azonnal eljutnak minden elérhető résztvevőhöz.

A backend egyik alapvető funkciója az üzenetküldés kezelése, amely egy dedikált HTTP POST végponton keresztül valósul meg. Ez a folyamat lehetővé teszi, hogy a felhasználók szöveges üzeneteket és opcionálisan fájlcsatolmányokat küldjenek egymásnak egy adott beszélgetésen belül.



. ábra Üzenet küldés api

A végpont működése több lépésben történik. Elsőként a szerver fogadja és feldolgozza a felhasználó által küldött adatokat, amelyeket validál, például az üzenet hossza alapján. A rendszer biztosítja, hogy a túl hosszú üzenetek ne kerülhessenek feldolgozásra, és ilyen esetben hibaüzenettel válaszol.

Amennyiben az üzenethez csatolmány is tartozik, a szerver azt külön menti el egy kapcsolódó adatbázistáblában, majd összekapcsolja az adott üzenettel. Az üzenet ezt követően elmentésre kerül az adatbázis megfelelő táblájába, ahol a kapcsolódó metaadatokat – például a feladó azonosítóját, a beszélgetés azonosítóját, az esetleges válaszüzenet-referenciát és a csatolmány azonosítóját – szintén rögzíti a rendszer.

Az adatok mentését követően a szerver lekérdezi a beszélgetés többi résztvevőjét, és azonosítja azokat a felhasználókat, akiknek a kliensoldali felületén meg kell jeleníteni az új üzenetet. Végül a szerver egy strukturált válaszüzenetben visszajelzést küld a kliensnek a művelet sikerességéről, illetve hiba esetén annak részleteiről.

Ez a végpont a kliens által megadott felhasználóazonosító (userId) alapján visszaadja a felhasználó beszélgetéseihez tartozó olvasatlan üzenetek számát. A végpont célja, hogy valós idejű frissítés során információt szolgáltasson arról, hogy mely beszélgetések tartalmaznak új, még nem olvasott üzeneteket.



. ábra Olvasatlan üzenetek api algoritmus

A lekérdezés három fő célt szolgál:

* Szűrés: A lekérdezés csak azokra a beszélgetésekre tér ki, amelyekhez a megadott felhasználó tartozik, és ahol legalább egy nem tőle származó üzenet nincs még megjelölve olvasottként (state != 'seen').
* Csoportosítás: Az eredményeket beszélgetésenként (conversationId) és annak típusánként (isGroupChat) csoportosítja.

## Tesztelés

A weboldal fejlesztését dinamikus tesztelési módszerekkel támogattuk. Ez azt jelentette, hogy a funkciók implementálását követően azonnal, az éles környezethez hasonló körülmények között teszteltük az egyes komponenseket. Ez a megközelítés lehetővé tette a hibák gyors azonosítását és kijavítását, valamint a fejlesztési folyamat rugalmas alakítását a felmerülő problémák függvényében.

Az egyik legjelentősebb probléma a socket-alapú kommunikációval kapcsolatos volt: az üzenetek nem mindig kerültek elküldésre vagy nem töltődtek be megfelelően, és az új funkciók integrálása is nehézségekbe ütközött. A problémák hátterében a meglévő socket rendszer megbízhatatlan működése állt. Ennek orvoslása érdekében a socket rendszert teljesen újraírtuk, így az üzenetek mostantól valós időben, megbízhatóan jutnak el minden felhasználóhoz. Az új architektúra emellett lehetővé tette további funkciók – például üzenetek törlése és szerkesztése – egyszerű hozzáadását is.

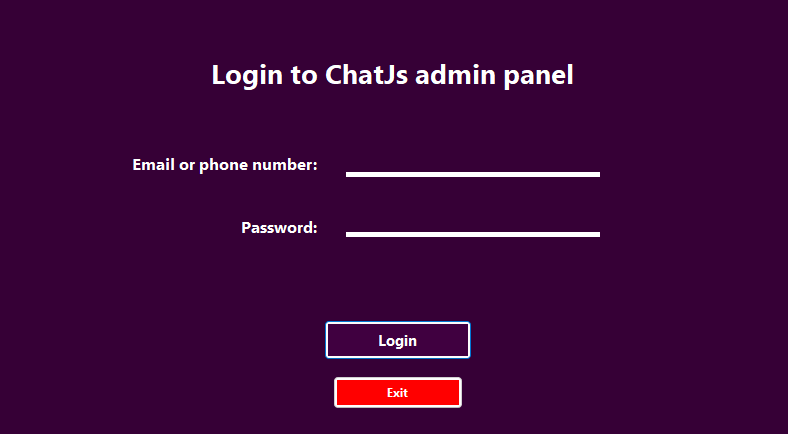
Egy másik gyakori hiba a belépési folyamat során jelentkezett: előfordult, hogy az első bejelentkezés után hibaüzenet jelent meg, amely csak az oldal újratöltésével tűnt el. Emellett probléma volt azzal is, hogy ha a felhasználó kijelentkezett, majd egy másik felhasználóval újra belépett, a rendszer mégis az előző felhasználó adatait jelenítette meg. Ezeket a problémákat végül a süti-kezelés helytelen időzítésére vezettük vissza, amelyet az érintett függvények áthelyezésével sikerült megoldani.

Néhány ismert probléma ugyanakkor még nem került teljes mértékben kijavításra. Az egyik ilyen a valós idejű üzenetküldés sorrendjéhez kapcsolódik: ha a felhasználó egy újabb üzenetet küld el, mielőtt az előző teljesen feldolgozásra kerülne – például lassú hálózati kapcsolat vagy nagy adatforgalom esetén –, előfordulhat, hogy az új üzenet technikailag megelőzi az előzőt. Ennek következtében az üzenetek megjelenítési sorrendje felborulhat: a rendszer nem feltétlenül a ténylegesen utolsó üzenetet jeleníti meg, hanem azt, amelyik utolsóként érkezett az adatbázisba.

# Felhasználói dokumentáció

## ChatJS asztali alkalmazás

Fontos megjegyezni, hogy az asztali alkalmazás kizárólag az oldal adminisztrátorai számára készült, és nem érhető el a hétköznapi felhasználók számára. Az alkalmazás megnyitását követően a felhasználót a bejelentkezési felület fogadja. Itt lehetőség van e-mail-cím vagy telefonszám, valamint jelszó megadására. A „Bejelentkezés” gombra kattintva a rendszer ellenőrzi a megadott adatokat, és sikeres hitelesítés után a felhasználó a főmenübe jut.



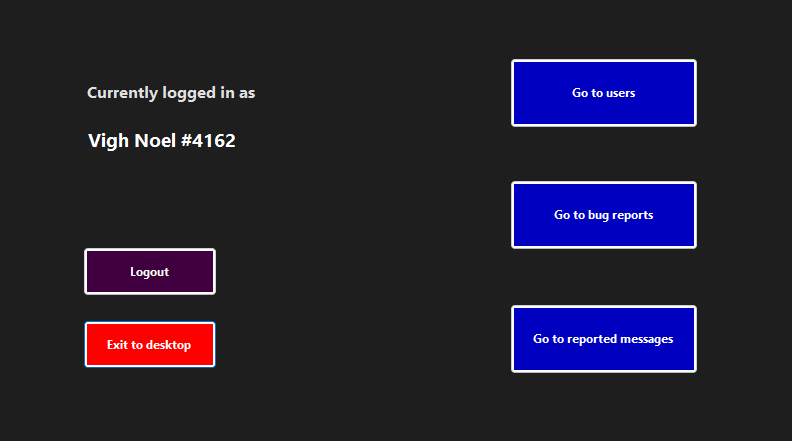
. ábra Bejelentkezés felület

A főmenü felületén a bal felső sarokban jelennek meg a bejelentkezett adminisztrátor adatai: a megjelenített név és az azonosító. A bal alsó sarokban két gomb található:

* **Kijelentkezés**: ezzel a gombal a felhasznaló visszaléphet a bejelentkezés ablakra
* **Kilépés**: Ezzel a gombbal a teljes alkalmazás bezárható

A jobb oldali oldalsávban három fő funkciógomb helyezkedik el, felülről lefelé haladva:

* **Felhasználók**
* **Hibajelentések**
* **Jelentett üzenetek**



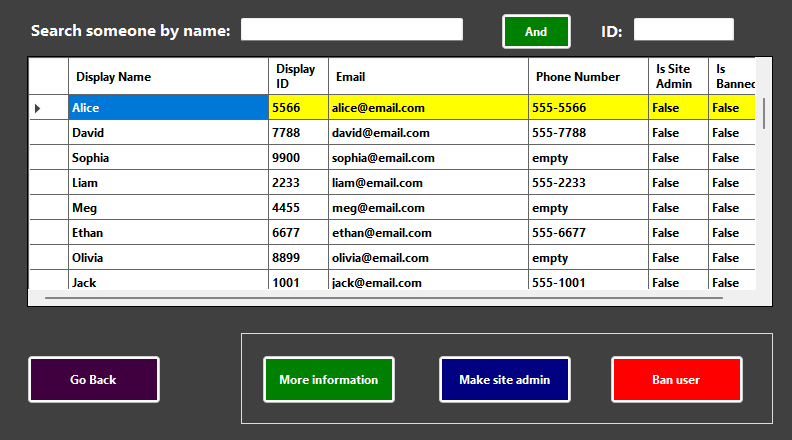
14. ábra Főmenü

**Felhasználók kezelése**

A „Felhasználók” gombra kattintva megnyílik a felhasználói adatokat tartalmazó ablak, ahol minden regisztrált felhasználó adatai megtekinthetők: a megjelenített név, azonosító, e-mail-cím, telefonszám, valamint az, hogy rendelkezik-e adminisztrátori jogosultsággal, illetve hogy ki van-e tiltva az oldalról.

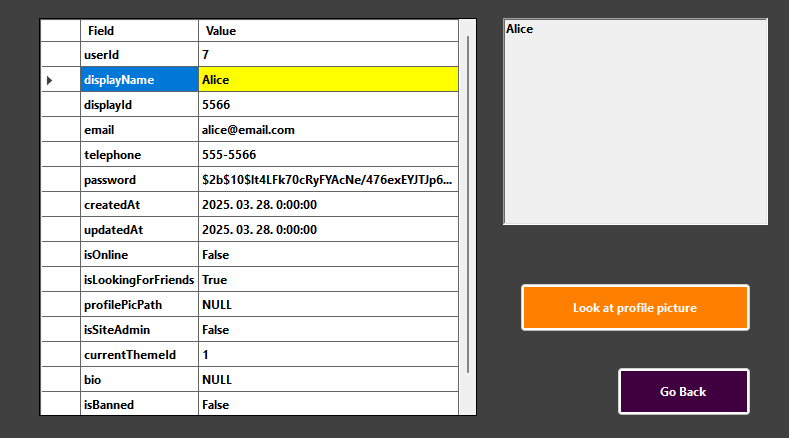
Az ablak tetején található keresősáv segítségével lehetőség van név és/vagy azonosító alapján történő szűrésre, a zöld gombra kattintva válthatunk a két opció között. Egy adott felhasználóra kattintva a képernyő alján egy vezérlőmenü jelenik meg, ahol a kiválasztott személy adminná tehető vagy kitiltható, illetve a további információkat is meg lehet nyitni.

A bal alsó sarokban elhelyezett Vissza gomb segítségével visszatérhetünk a főmenübe.



15. ábra Felhasználók ablakl

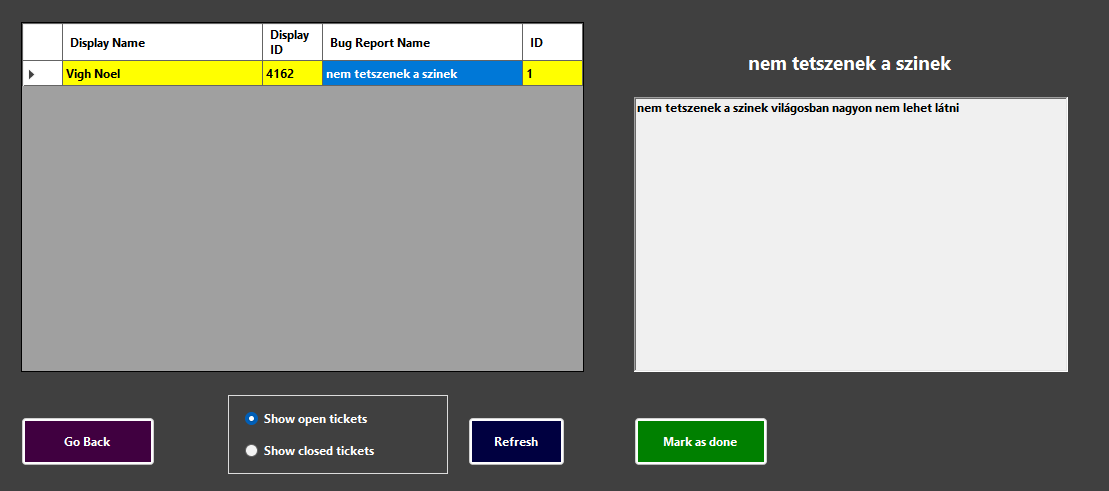
A “Több információ” gombra kattintva megnyílik egy új ablak a felhasználó részletes informácóival, ahol pontosabb leírást kaphatunk az adott felhasználóról. A “Profilkép megtekintése” gombra kattintva lehetőségünk van a felhasználó profilképét megnézni, a ”Vissza” gombbal pedig vissza jutunk az előző ablakhoz.



16. ábra Több információ ablak

**Hibajelentések kezelése**

A főmenü második opcióját, a „Hibajelentések” gombot választva megnyílik a hibajelentések ablaka. Itt megtekinthetők az aktív hibajelentések, amelyeket a beküldő neve, azonosítója, a jelentés címe és azonosítója alapján azonosíthatunk. Egy jelentésre kattintva megjelenik annak részletes szöveges tartalma.

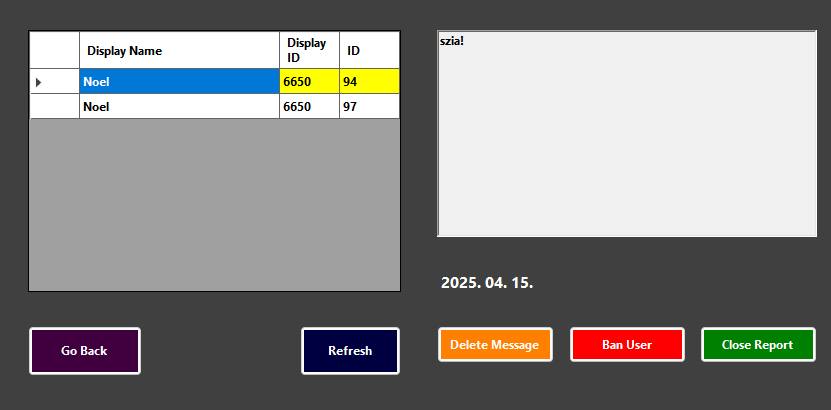


17. ábra Hibajelentések ablak

Az ablak alsó részén lehetőség van szűrni a jelentéseket állapot szerint (nyitott vagy lezárt), míg a jobb oldalon található Frissítés gomb segítségével a megjelenített adatok naprakészen tarthatók. A Vissza gomb itt is a bal alsó sarokban érhető el.

**Jelentett üzenetek kezelése**

A főmenü harmadik lehetősége a „Jelentett üzenetek” menüpont. Erre kattintva megnyílik az erre dedikált ablak, ahol a jelentett üzenetek listája jelenik meg. Minden bejegyzésnél látható az üzenet küldőjének neve és azonosítója, valamint az üzenet egyedi azonosítója. Egy jelentésre kattintva megtekinthető az adott üzenet tartalma is.

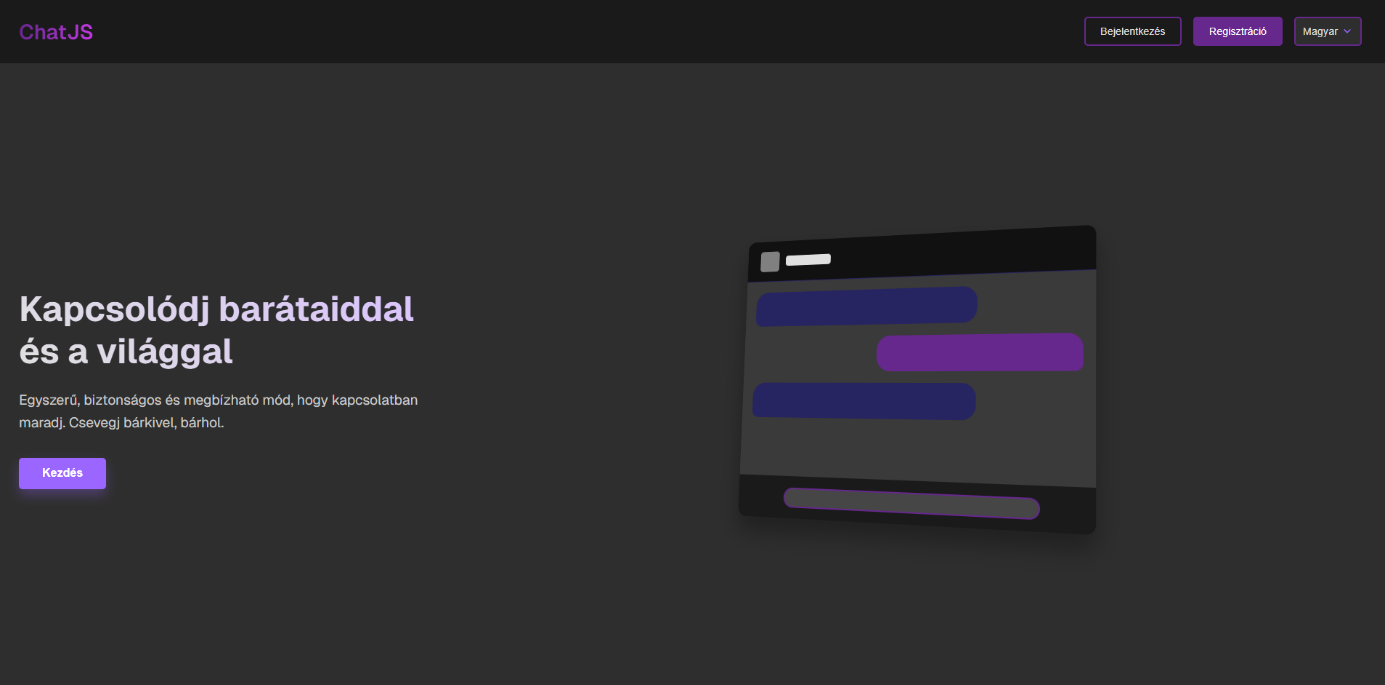


18. ábra Jelentett üzenetek ablak

Az ablak alján található a Frissítés gomb, míg a bal alsó sarokban a Vissza a főmenübe gombbal lehet navigálni.

## ChatJS webalkalmazás

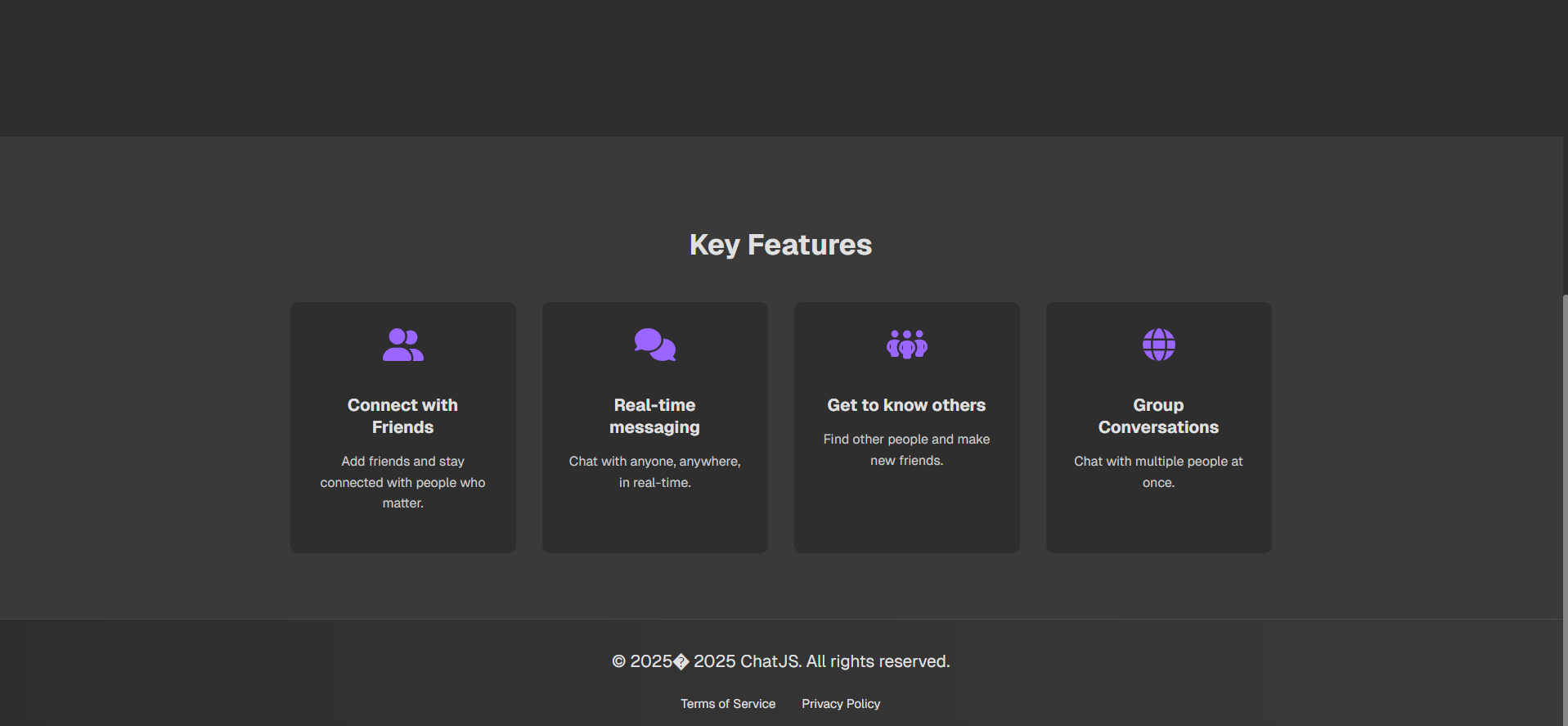
A felhasználó először egy landoló oldalra érkezik, amely az oldal bemutatkozó és tájékoztató felülete. Itt lehetősége van megismerkedni az oldal előnyeivel és funkcióival. Amennyiben a felhasználó úgy dönt, hogy érdeklődik a szolgáltatás iránt, a képernyő közepén elhelyezkedő kezdés gomb segítségével elindíthatja a regisztrációs folyamatot. Fontos megjegyezni, hogy az oldal regisztráció nélkül nem használható, és a regisztráció lehetőségét a jobb felső sarokban található Regisztráció gombra kattintva is elérhetjük.



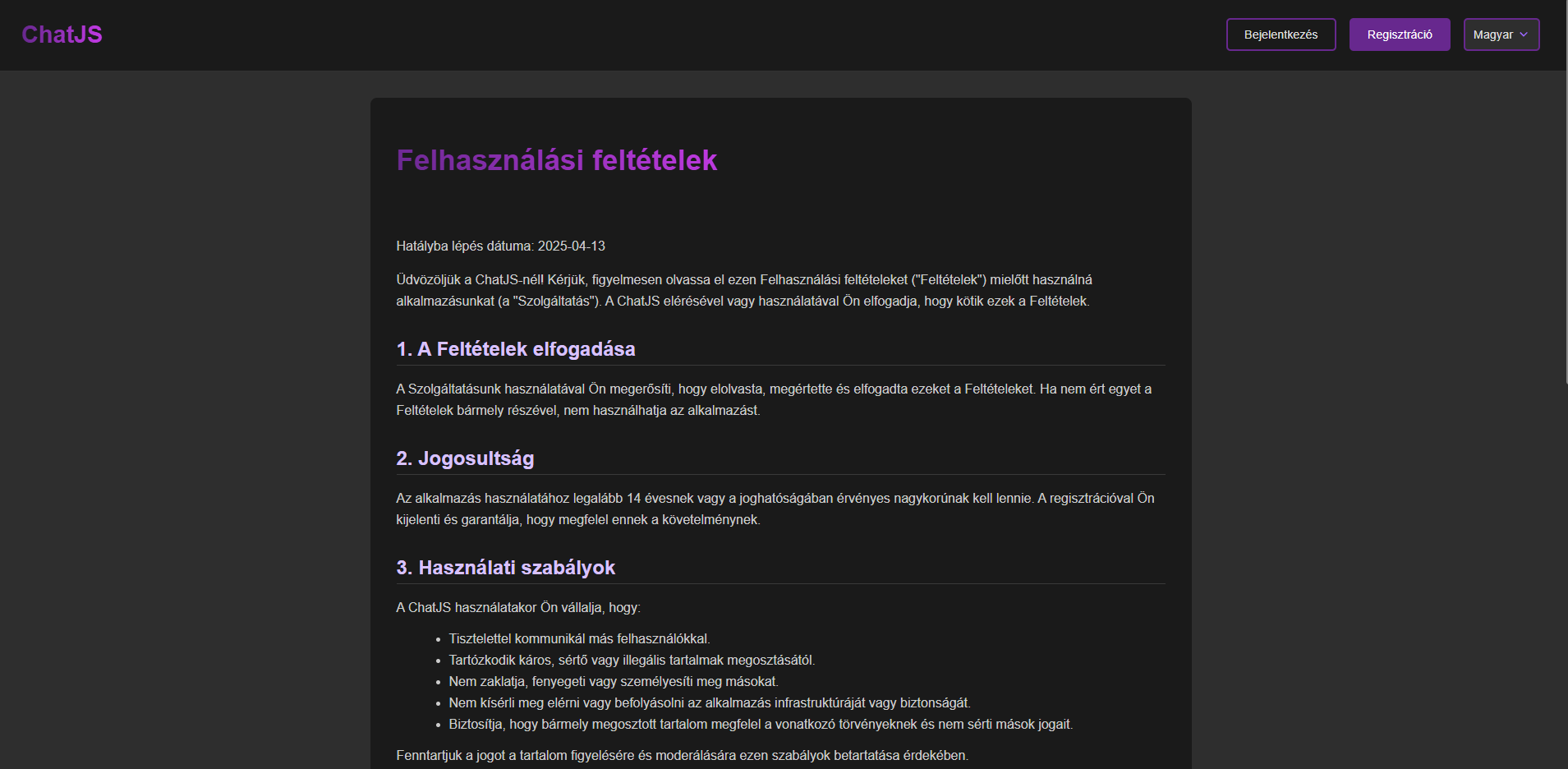
19. ábra ChatJS landoló oldal

A nyitóoldal jobb felső sarkában található a regisztráció gomb, mellett bal oldalon helyezkedik el a bejelentkezés gomb. Utóbbi lehetőséget biztosít a már regisztrált felhasználók számára a rendszerbe való belépésre. A jobb oldalon egy lenyíló menü is elérhető, amelyben két nyelv – magyar és angol – közül választhatunk. Ez a beállítás az oldal nyelvét határozza meg.

Az oldal alján található egy rövid bemutatkozó szöveg a platformról, valamint a láblécben két fontos hivatkozás: a Felhasználási feltételek és az Adatvédelmi irányelvek. Ezekre kattintva a felhasználó az adott tartalmakat tartalmazó külön oldalra navigálódik. Mindkét oldalon a bal felső sarokban megtalálható a platform neve (ChatJS) mint logó és egyben hivatkozás, amelyre kattintva vissza irányítódunk a nyitóoldalra.



20. ábra Landoló oldal alja

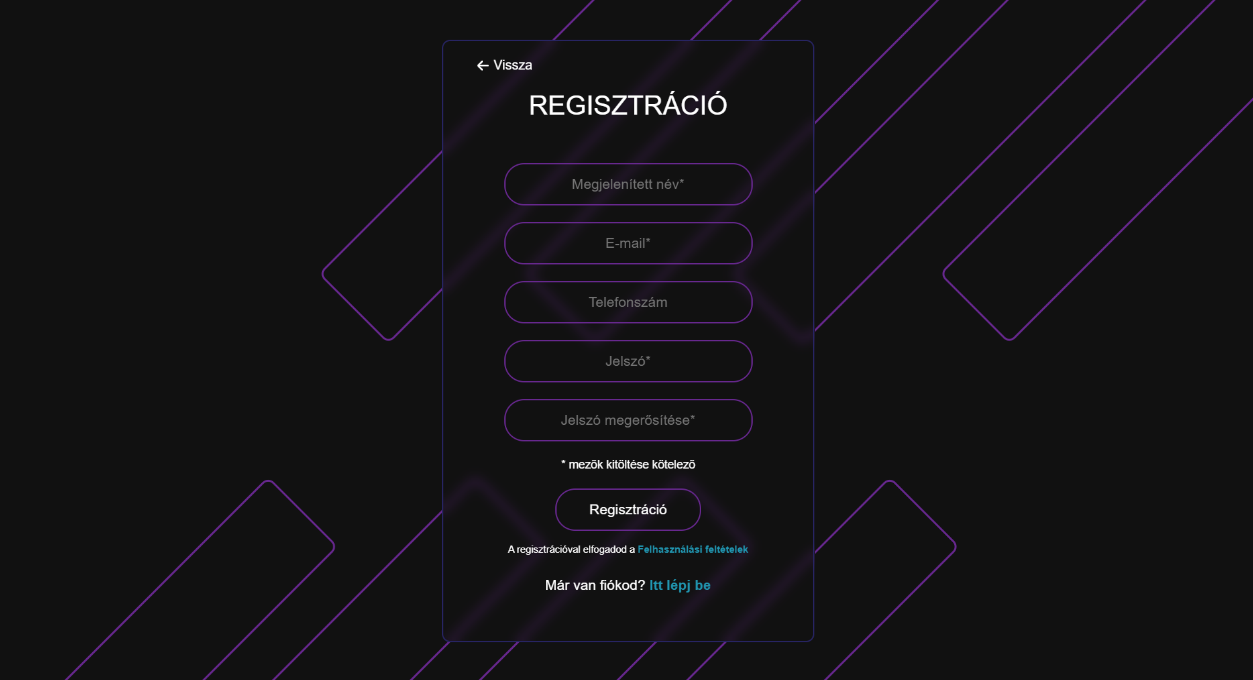


21. ábra Felhasználási feltételek

A Regisztráció gombra kattintva kezdődik meg az új fiók létrehozása. A folyamat során az alábbi mezők kitöltése szükséges (a \*-gal jelölt mezők kötelezőek):

* Megjelenített név\*: minimum 4 karakter hosszú.
* E-mail cím\*: egyedi azonosítóként szolgál, kötelezően megadandó.
* Telefonszám: opcionálisan megadható.
* Jelszó\*: minimum 8 karakterből áll, tartalmaznia kell legalább egy nagybetűt és egy számot.
* Jelszó megerősítése\*: a jelszó ismételt megadása az egyezőség ellenőrzésére.

A szükséges adatok megadása után a Regisztráció gombra kattintva véglegesíthetjük a fiók létrehozását. Regisztrációkor a felhasználó automatikusan elfogadja a Felhasználási feltételeket (21. ábra). Amennyiben a felhasználónak már van fiókja, az oldal alján található "Itt lépj be" feliratra kattintva közvetlenül átléphet a bejelentkezési felületre. Sikeres regisztráció esetén a rendszer automatikusan tovább irányít a bejelentkezési oldalra.

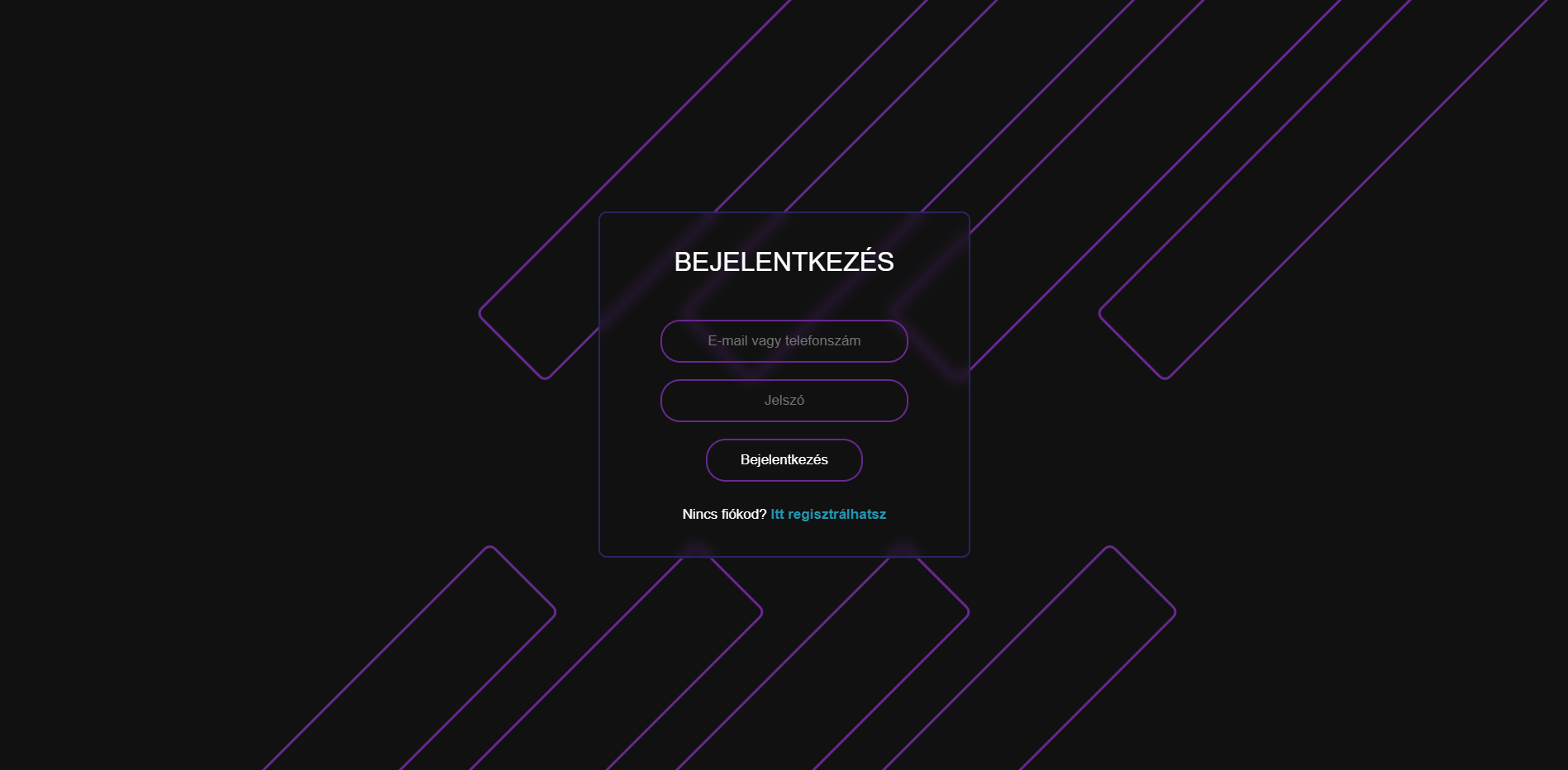


22. ábra Regisztrációs oldal

A bejelentkezési oldalon két adat megadása szükséges:

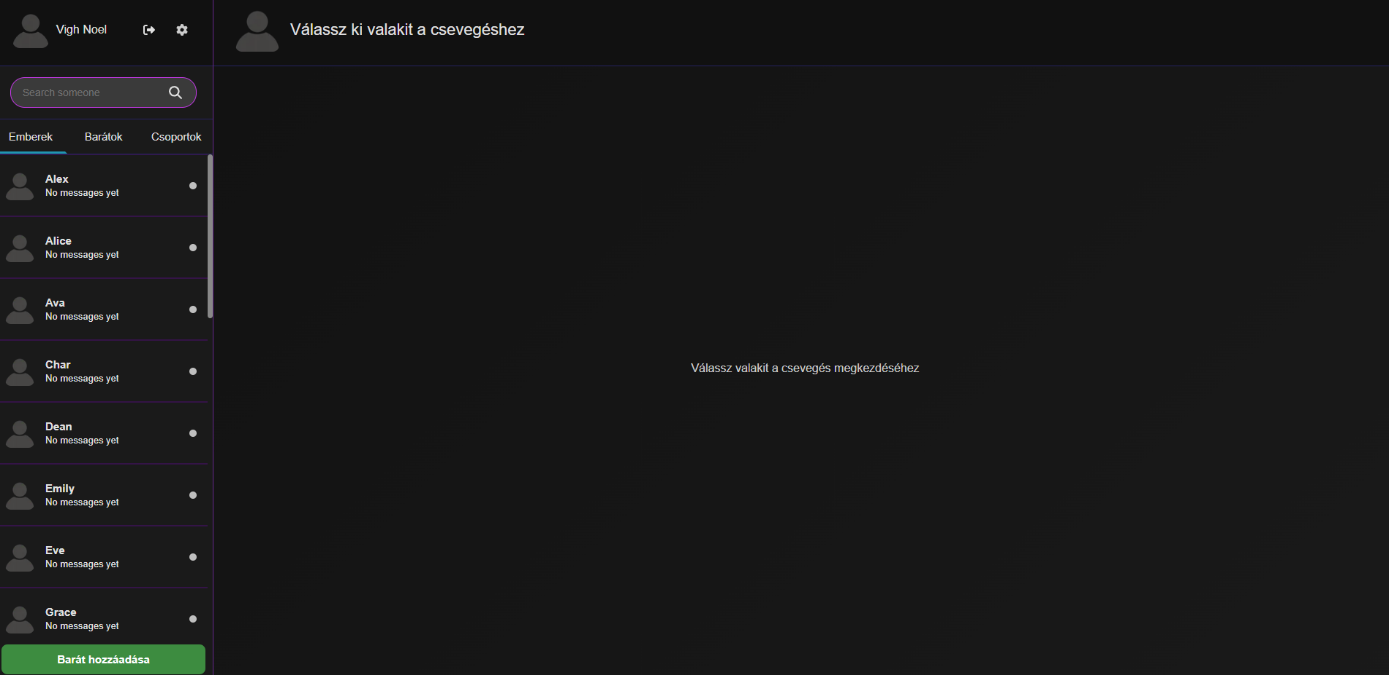
* E-mail cím vagy telefonszám
* Jelszó

A Bejelentkezés gombra kattintva a rendszer hitelesíti a megadott adatokat, és sikeres bejelentkezés után a felhasználó a chatfelületre kerül. Ha a felhasználó még nem rendelkezik fiókkal, az oldal alján található "Itt regisztrálhatsz" linkre kattintva elérheti a regisztrációs űrlapot.



23. ábra Bejelentkezés oldal

A következőkben bemutatom a chat felület elrendezését.



24. ábra Chat felület

Bejelentkezés után a felhasználó a fő chatfelületre jut. A bal oldalon egy átfogó menü található, amely több funkciót is ellát:

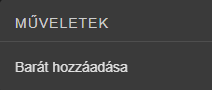
1. Profil:

A menü tetején a felhasználó profilképe és megjelenített neve jelenik meg. Mellette található a Kilépés gomb a fiókból való kijelentkezéshez és az oldal elhagyásához. A jobb oldalon található a Beállítások gomb, amely a beállítási menübe navigál (ennek részletes bemutatása a bal oldali menü bemutatása után szerepel).

1. Keresőmező

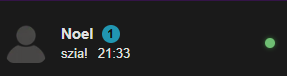
A keresőmező segítségével ismerősökre, felhasználókra vagy csoportokra kereshetünk név alapján.

1. Navigációs menüpontok
   1. Emberek: Itt azok a felhasználók jelennek meg, akik nyitottak új kapcsolatokra. Csak azokkal lehet üzenetet váltani, akik szintén keresnek barátokat. Jobb kattintással barátkérelem küldhető.



. ábra Jobb kattintásos menü az emberek menüpontban

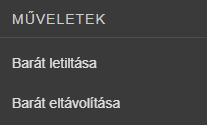
* 1. Barátok: Az elfogadott barátok listája. Itt Megjelenik a név mellette az olvasatlan üzenetek száma ha vannak, alatta az utolsó üzenet és annak ideje, illetve a barát státusza (zöld: elérhető, szürke: nem elérhető).



. ábra Példa egy barátról

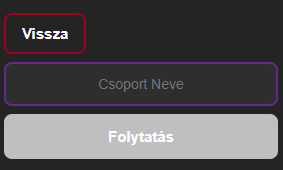
Jobb kattintással lehetőség van:

* + Letiltásra: az adott barát nem tud többé üzenetet küldeni, de a listán marad és a beszélgetés továbbra is megtekinthető marad.
  + Törlésre: teljes törlés a barátlistából, de továbbra is elérhető az Emberek menüpontban.

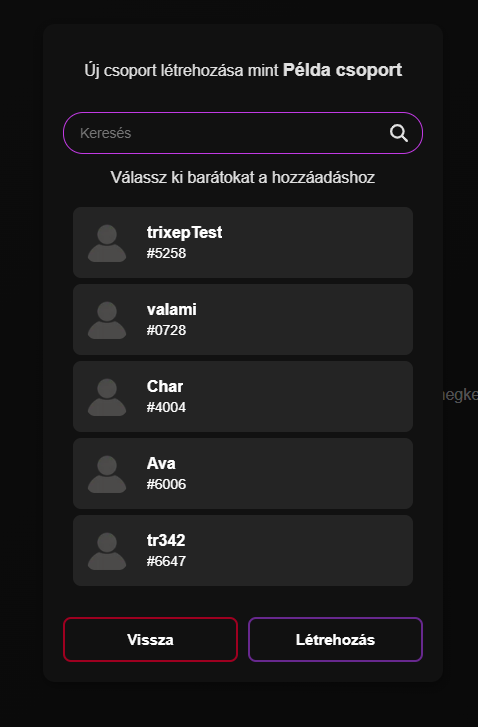


. ábra Jobb kattintásos menü a barátok menüpontban

* 1. Csoportok: Itt csoportos beszélgetések hozhatók létre. Az alján található Új csoport létrehozása gombbal megadható a csoport neve, majd kiválaszthatók a hozzáadni kívánt barátok. A csoporthoz csak barátok adhatók hozzá.

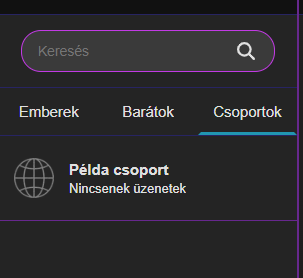


. ábra Csoport létrehozása 1



. ábra Csoport létrehozása 2

Létrehozáskor bal oldalt, a csoportok menüpont alatt megjelenik a létrehozott csoport.



. ábra Létrehozott csoport

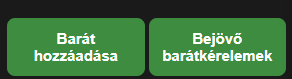
1. Barát hozzáadása

* Ez a gomb lehetőséget ad olyan felhasználók bejelölésére is, akik nem keresnek aktívan barátokat. A megjelenített név és az egyedi azonosító (ID, pl. 1234) megadása után küldhető el a barátkérelem.

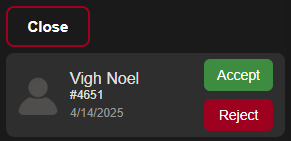
1. Bejövő barátkérelmek

Amennyiben van a felhasználónak aktív barátkérelme a barát hozzáadása gomb mellet megjelenik a bejövő barátkérelmek gomb. Ezt a menüt megnyitva megjelennek a kérelmek, ki írja a küldő nevét alatta az azonosítóját, illetve a kérelem elküldésének időpontját, a jobb oldalon két lehetőség áll rendelkezésre:

* Elfogadás: elfogadást követően felhasználó automatikusan megjelenik a Barátok listában és azonnal lehet is beszélgetni.
* Elutasítás: a kérelem törlésre kerül.

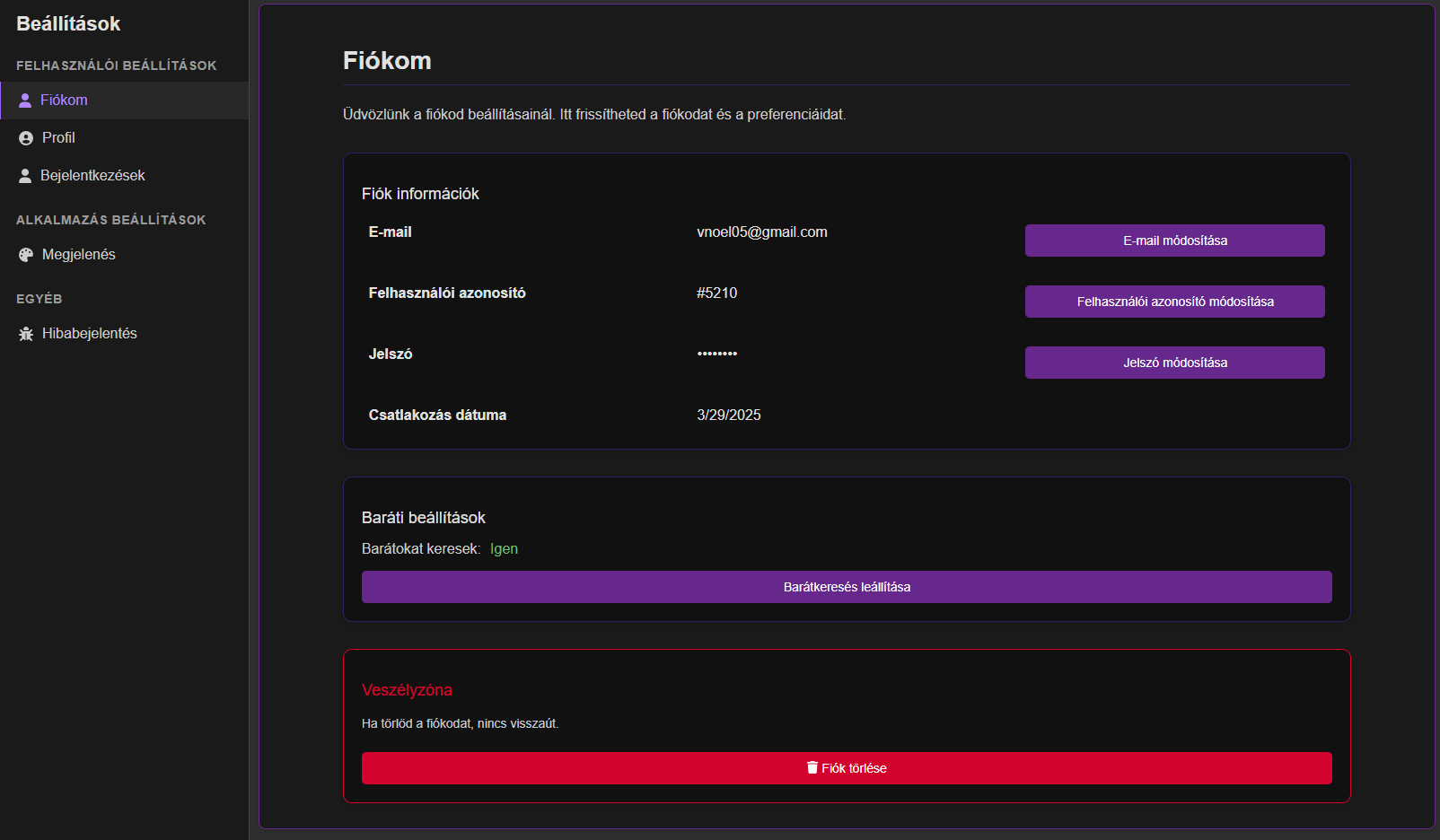


31. ábra Barát gombok



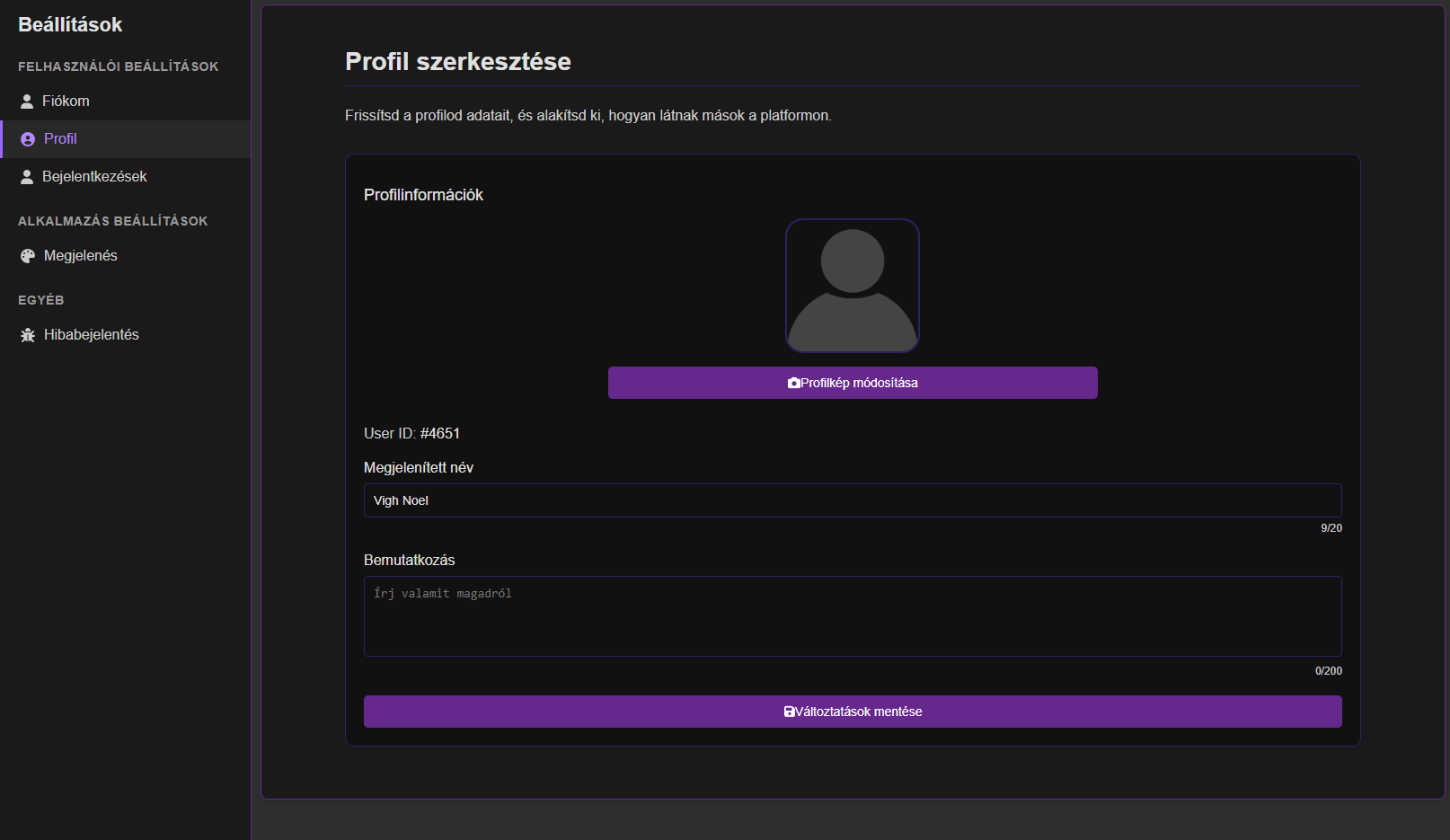
32. ábra Példa egy barátkérelemre

1. Beállítások: a beállításokat megnyitva (fogaskerék az oldal tetején) elsőnek a fiókom menüponttal találkozhatunk



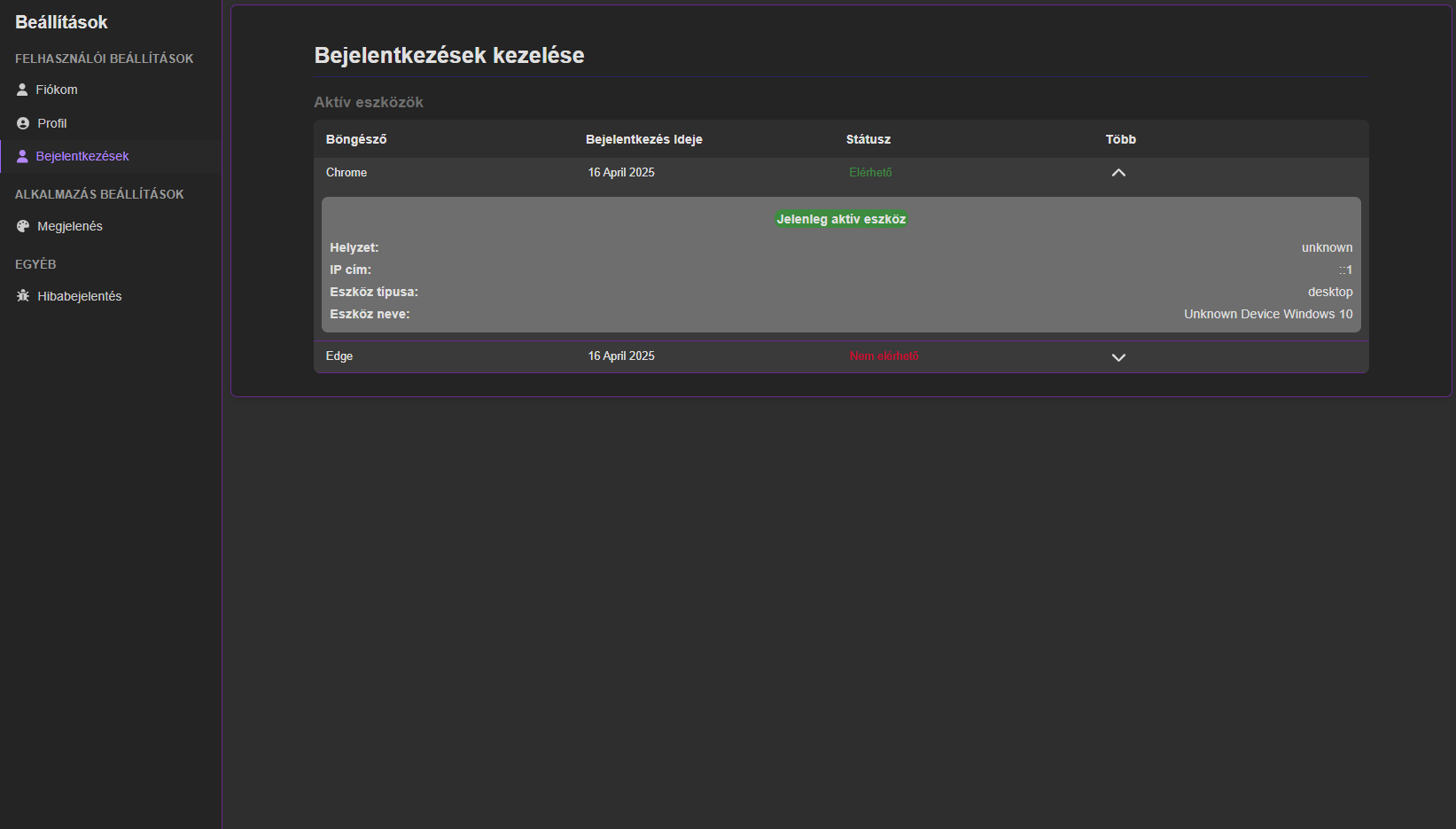
33. ábra Beállítások: Fiókom menüpont

Ebben a menüpontban találhatóak a legfontosabb beállítások. Itt megváltoztatható az e-mail cím, felhasználói azonosító és a jelszó a megfelelő gombra kattintva. Lejjebb a baráti beállítások szekcióban megadhatjuk, hogy szeretnénk-e barátokat keresni, vagy sem. A menüpont alján pedig a „Veszélyzóna” található, ahol a „Fiók törlése” opcióval, egy megerősítést követően, a felhasználó törölheti a profilját.

A második menüpont a „Profil”, ahol azok az adatok jelennek meg, amelyeket a másik felhasználók láthatnak.

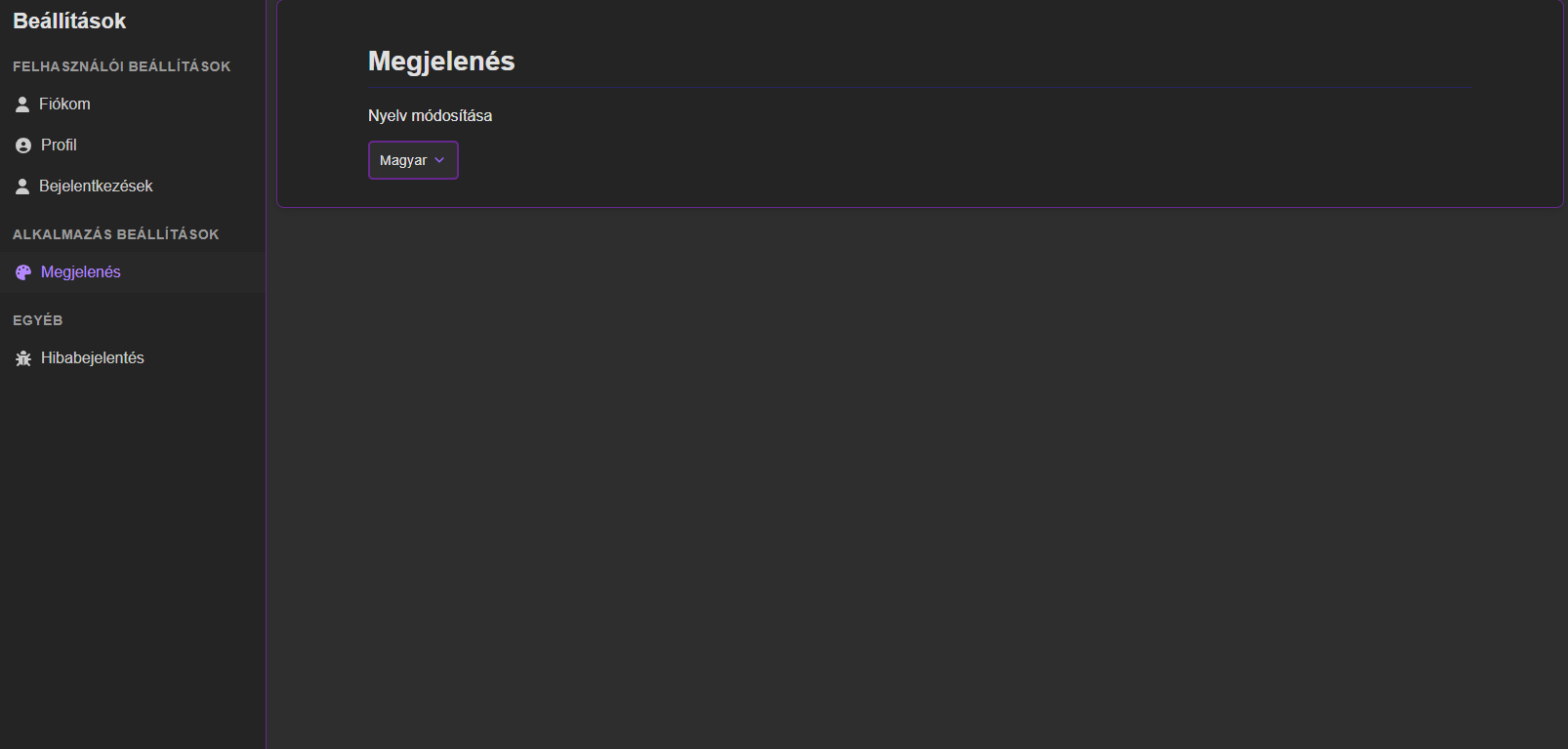
34. ábra Beállítások: Profil menüpont

A Profil menüpont alatt azokat az adatokat lehet módosítani, amiket más felhasználók számára is látható. A profilkép módosítása gombra kattintva a felhasználó feltölthet egy saját profilképet az oldalára, amelyet más felhasználók is láthatnak. A kép alatt módosítható a megjelenített név, amelynek hossza minimum 4, maximum 20 karakter lehet. Ezt követi a „Bemutatkozás” mező, ahol a felhasználó egy rövid, legfeljebb 200 karakteres bemutatkozást írhat. Fontos megjegyezni, hogy a változtatások csak akkor kerülnek mentésre, ha a „Változtatások mentése” gombra kattintanak. Ha a felhasználó korábban elhagyja az oldalt, a változtatások elvesznek.

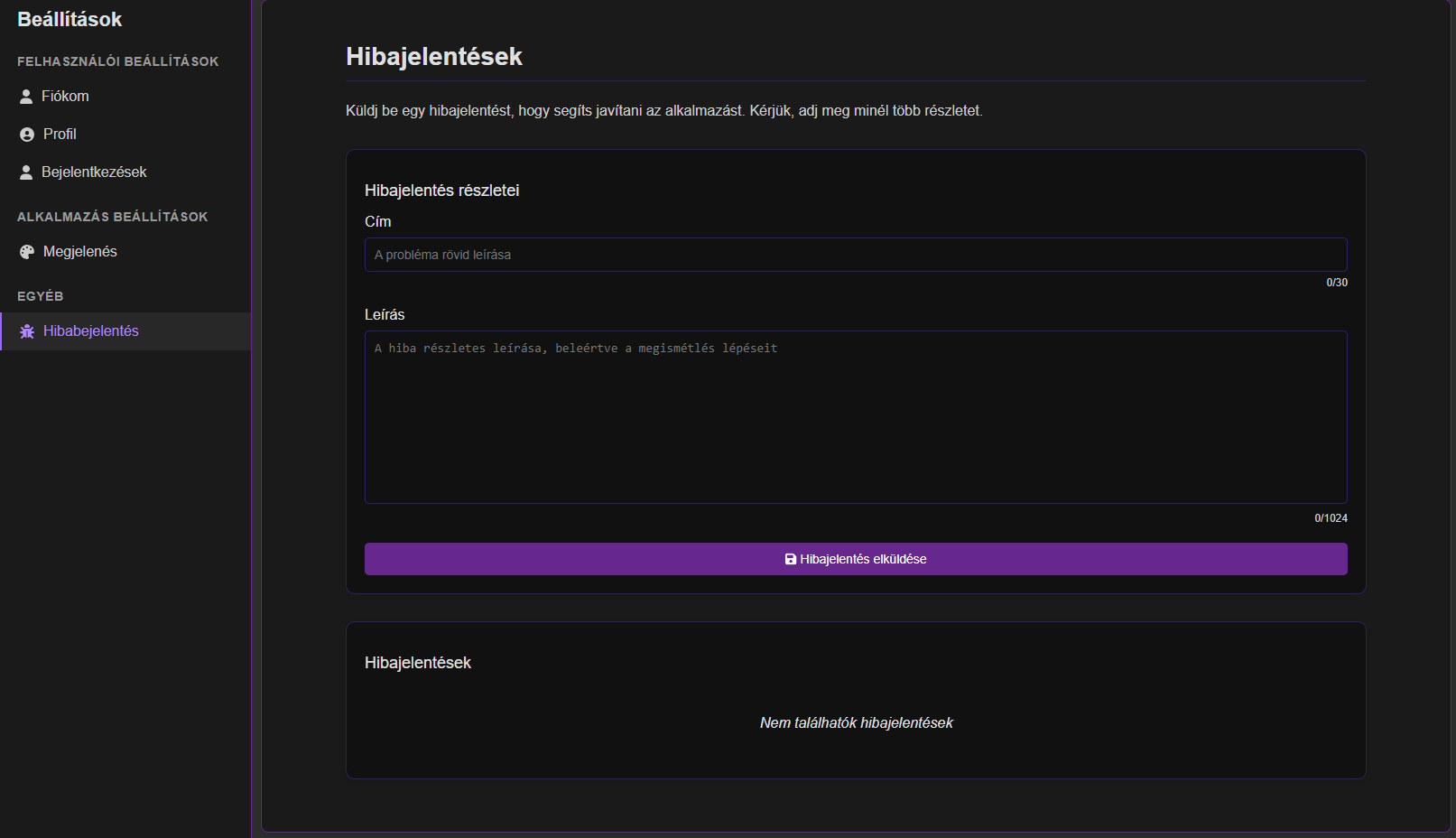
A következő menüpont a „Bejelentkezések”, ahol a korábbi, még bejelentkezett eszközeinket tudjuk megtekinteni. A menüben először egy rövid leírást látunk az eszközről, a böngészőt, a bejelentkezés idejét és az elérhetőségi státuszt. A „Több” gombra kattintva megjelenik a többi részlet: a helyzet, IP cím, eszköz típusa és neve. Középen zölden jelöli, ha az adott eszköz amit pont nézünk az az eszköz-e, amin jelenleg be vagyunk jelentkezve.

. ábra Bejelentkezések menüpont

A következő menüpont a „Megjelenések”, ahol az oldal nyelvét lehet állítani, illetve a színsémát is lehetett volna, de sajnos azt nem tudtuk elkészíteni *(lásd később: Továbbfejlesztési lehetőségek)*. Az elérhető nyelvek a magyar, illetve az angol. Kiválasztáskor azonnal frissíti az oldal a szöveget a kiválasztott nyelvre.



. ábra Megjelenés menüpont

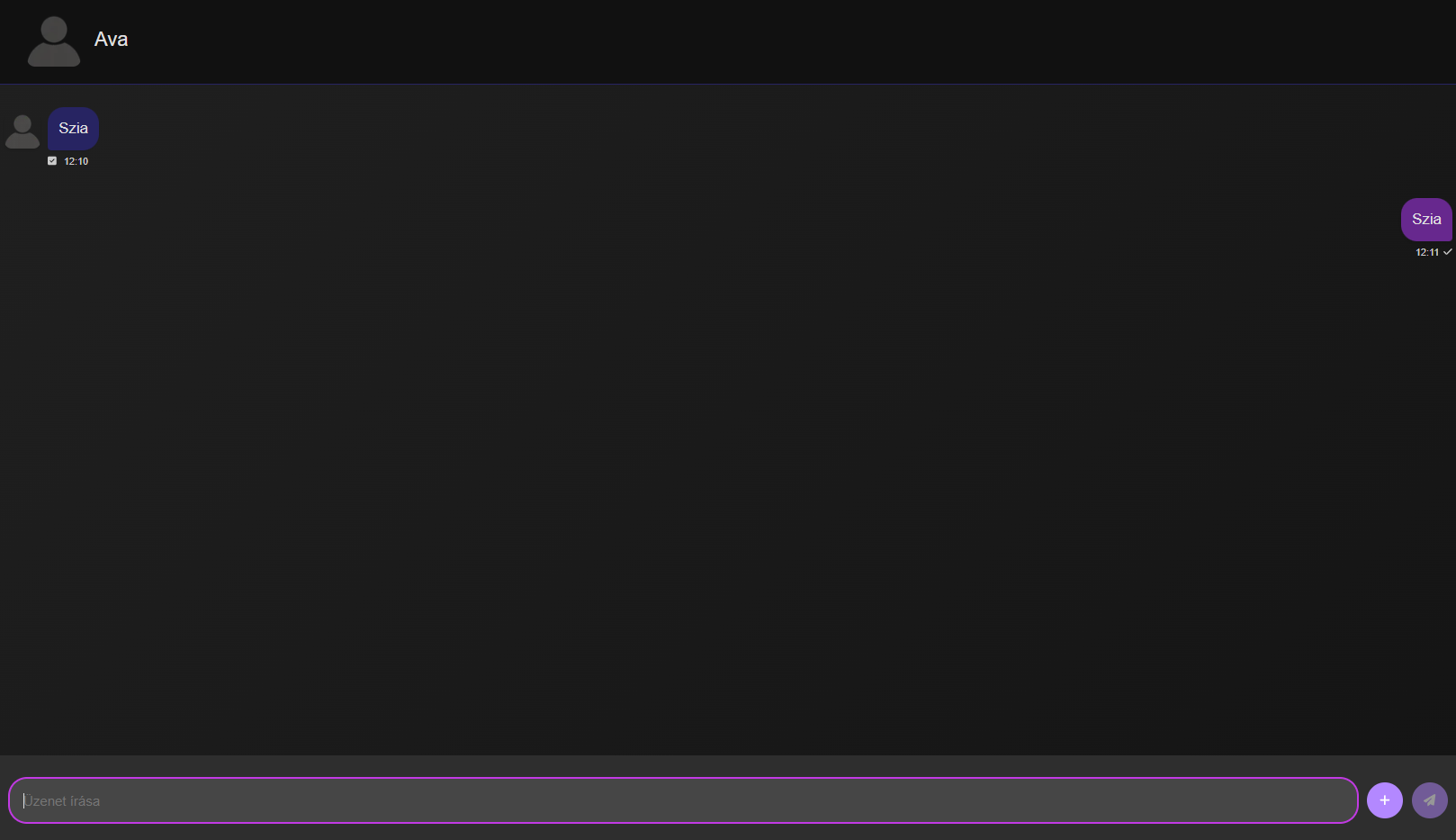
Az utolsó menüpont a „Hibajelentések”, ahol a felhasználó értesíthet minket, ha egy hibát talál az oldalon.

37. ábra Beállítások: Hibajelentések menüpont

A hibajelentés címének megadása kötelező, mivel ez rövid betekintést nyújt a problémába. A „Leírás” mezőben részletesen ki kell fejteni a hibajelentést, beleértve a hiba előhozásának menetét is. Opcionálisan, egy képet is lehet feltölteni a hibáról, ami segítheti a hiba megismerését. Az „Elküldés” gombra kattintva a hibajelentést rögzítjük, amely ezt követően megjelenik a hibajelentések szekció alján. Itt a felhasználó láthatja a hibajelentés állapotát, például, hogy lezárásra került-e.

A chat felület:

A fő felület, ahol a beszélgetéseket láthatjuk miután rákattintottunk egy emberre vagy csoportra

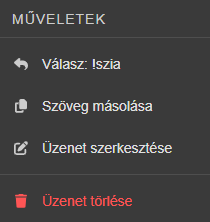


A menü tetején látható annak a felhasználónak a neve és profilképe, akire rákattintottunk. Középen jelennek meg az üzenetek, azonban, ha nincsenek üzenetek, akkor a „Még nincsenek üzenetek” felirat látható. Alul található az input mező, ahová írva üzenetet küldhetünk a felhasználónak, az Enter billentyű vagy a jobb oldalon található gomb megnyomásával.

Egy üzenetre kattintva két lehetőség áll rendelkezésre:

1. Ha a saját üzenetünkre jobb gombbal kattintunk akkor a megjelenő menüben a következő opciók közül választhatunk:

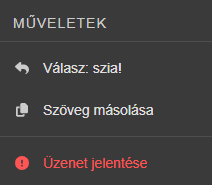
* Válasz – ezzel az opcióval válaszolhatsz az adott üzenetre;
* Szöveg másolása – lehetővé teszi a szöveg kimásolását;
* Üzenet szerkesztése – az üzenetet módosíthatjuk, azonban a szerkesztés után az „Szerkesztve” felirat jelenik meg a szöveg alatt, amely jelzi a másik felhasználó számára, hogy az üzenet módosítva lett;
* Üzenet törlése – az üzenet törlésével a „Törölt üzenet” felirat jelenik meg az üzenet helyén.



38. ábra Példa saját üzenetre kattintásról

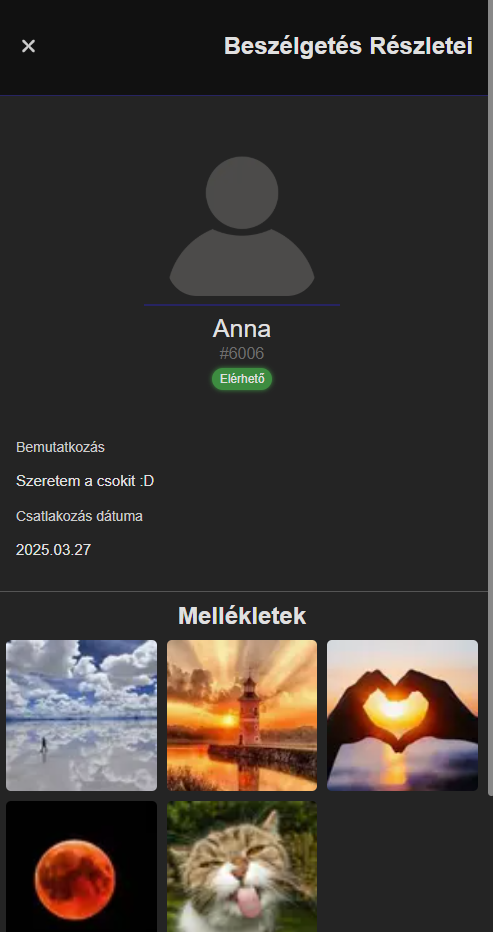
1. Ha a másik felhasználó üzenetére jobb gombbal kattintunk, a megjelenő menüben a következő lehetőségek közül választhatunk:

* Válasz – lehetőséget nyújt az adott üzenetre való válaszadásra;
* Másolás – az üzenet szövege kimásolható;
* Üzenet jelentése – ezzel az opcióval jelentheted az üzenetet, ha úgy véled, hogy olyan tartalommal rendelkezik, amely sérti a felhasználási feltételeket.



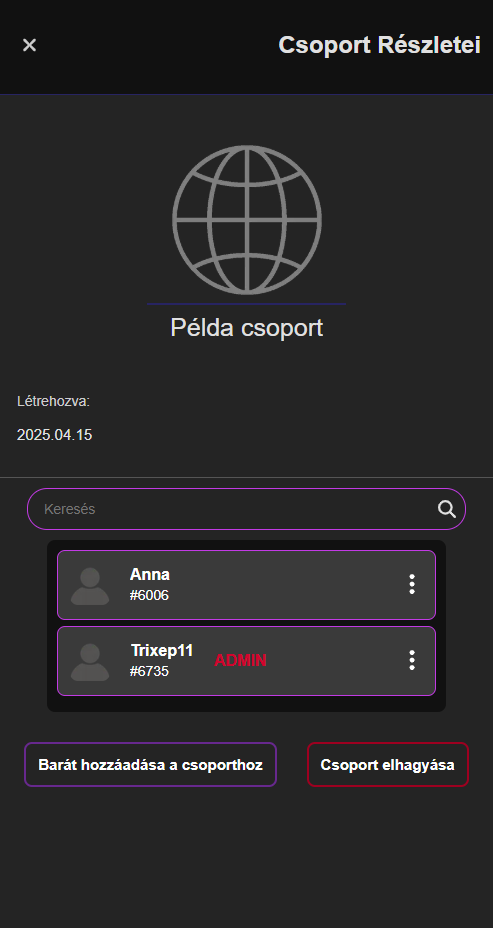
39. ábra Példa Mások üzenetére kattintásról

Az oldal tetején a névre kattintva megnyitható a további információk panel, amelyen látható a másik felhasználó adatai, például a profilképe, az azonosítója, a neve, és a státusza. Továbbá megjelennek a beszélgetésben megosztott képek is. Ha nincsenek képek, akkor a „Nincsenek mellékletek” szöveg jelenik meg. Képet az input mező melletti „+” gombra kattintva választhatunk ki, de beilleszthetünk képet a vágólapról is. Alternatívaként, ha az input mezőre húzunk egy képet, az automatikusan feltöltésre kerül.



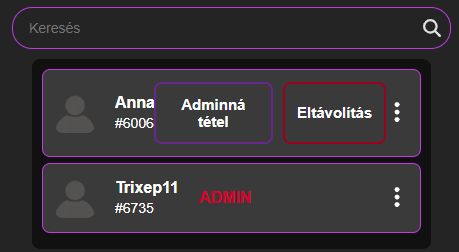
40. ábra Példa a további információk panelről

Ha egy csoportos beszélgetésben vagyunk, és úgy kattintunk a információk panelre, akkor a csoporthoz tartozó információk jelennek meg, illetve a csoportban lévő emberek. Ha mi vagyunk a csoport egyik adminja, akkor további lehetőség nyilik a résztvevők kezelésére is.



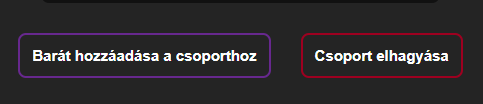
. ábra Csoport részletei

A csoporttagot adminná tudjuk tenni, így ő is képes lesz kezelni a csoporttagokat, ha valaki admin az el tudjuk venni másnak a jogusultságát, illetve z embereket el is tudjuk távolítani.



. ábra Csoporttag kezelése

A csoporthoz később is tudd bárki még barátot utólag hozzáadni, illetve bármikor ki is lehet lépni a csoportból.



. ábra Csoporthoz adás és kilépés

Ha az egyetlen admin kilép a csoportból, név szerint az első ember lesz az új admin. Ha mindenki kilép a csoportból, a csoport törlésre kerül.

# Továbbfejlesztési lehetőségek

* Elfelejtett jelszó és e-mail kezelés: Az elfelejtett jelszó funkció és az e-mail cím módosításának lehetősége olyan alapvető funkciók, amelyek nemcsak a felhasználói élményt javítanák, hanem a rendszer biztonságát is növelnék. Ezáltal a felhasználók könnyedén visszaállíthatnák hozzáférésüket a fiókjukhoz, valamint szükség esetén frissíthetnék személyes adataikat.
* Személyre szabható felület: A testre szabható felhasználói felület bevezetése egyéni vizuális preferenciák szerint lehetővé tenné a felhasználók számára, hogy a platformot saját ízlésüknek megfelelően alakítsák. Színvilág, témák és stíluselemek választási lehetősége hozzájárulna ahhoz, hogy mindenki számára kényelmesebbé váljon a használat, és növelné a platform személyességét.
* Videók küldése: A médiafájlok küldésének lehetősége fontos kiegészítő funkció lehetne, amely a személyes és üzleti kommunikációt egyaránt gazdagítaná. A képek és videók megosztásával a felhasználók interakciói sokkal dinamikusabbá, vizuálisan érdekesebbé válnának.
* Hang- és videóhívások: Az alkalmazás kiterjesztése valós idejű hang- és videóhívások funkcióval lehetővé tenné, hogy a felhasználók személyesebb és közvetlenebb kommunikációt folytathassanak egymással. Ez a fejlesztés nemcsak a felhasználói élményt javítaná, hanem a platform versenyképességét is növelné, mivel egyre nagyobb az igény a multimédiás interakciókat támogató megoldásokra.

# Összegzés

Projektünk célja egy olyan platform létrehozása volt, amely lehetővé teszi a felhasználók számára, hogy bárhonnan és bármikor egyszerűen kapcsolatba léphessenek másokkal. Ez volt az első nagyobb fejlesztési projektünk, amely során számos kihívással szembesültünk, különösen a csapatmunka és a határidők betartása terén.

A fejlesztés kezdeti szakaszában az új technológiák, például a Next.js és a Socket.IO használata során felmerülő nehézségek lassították a munkát, azonban ezek a problémák jelentős tanulási tapasztalatokkal jártak. Ennek ellenére sikerült megvalósítanunk egy működő prototípust, amely alapvető funkciókat kínál, mint például a valós idejű üzenetküldés és az egyszerű felhasználói felület.

# Ábrajegyzék

[1. ábra EER ábra a chatdb adatbázisról 13](#_Toc195699449)

[2. ábra Bejelentkezés függvénye 14](#_Toc195699450)

[3. ábra A jelentések betöltése algoritmus 16](#_Toc195699451)

[4. ábra Kép megjelenítése algoritmusok 17](#_Toc195699452)

[5. ábra Cell click algoritmus 19](#_Toc195699453)

[6. ábra Üzenet törlése algoritmus 21](#_Toc195699454)

[7. ábra Barát hozzáadás algoritmus 22](#_Toc195699455)

[8. ábra Státusz változás algoritmus 24](#_Toc195699456)

[9. ábra Idő formázása algoritmus 26](#_Toc195699457)

[10. ábra Csoport létrehozás socket 27](#_Toc195699458)

[11. ábra Üzenet küldés api 29](#_Toc195699459)

[12. ábra Olvasatlan üzenetek api algoritmus 31](#_Toc195699460)

[13. ábra Bejelentkezés felület 33](#_Toc195699461)

[14. ábra Főmenü 34](#_Toc195699462)

[15. ábra Felhasználók ablakl 35](#_Toc195699463)

[16. ábra Több információ ablak 35](#_Toc195699464)

[17. ábra Hibajelentések ablak 36](#_Toc195699465)

[18. ábra Jelentett üzenetek ablak 37](#_Toc195699466)

[19. ábra ChatJS landoló oldal 38](#_Toc195699467)

[20. ábra Landoló oldal alja 39](#_Toc195699468)

[21. ábra Felhasználási feltételek 39](#_Toc195699469)

[22. ábra Regisztrációs oldal 40](#_Toc195699470)

[23. ábra Bejelentkezés oldal 41](#_Toc195699471)

[24. ábra Chat felület 41](#_Toc195699472)

[25. ábra Jobb kattintásos menü az emberek menüpontban 42](#_Toc195699473)

[26. ábra Példa egy barátról 42](#_Toc195699474)

[27. ábra Jobb kattintásos menü a barátok menüpontban 42](#_Toc195699475)

[28. ábra Csoport létrehozása 1 43](#_Toc195699476)

[29. ábra Csoport létrehozása 2 43](#_Toc195699477)

[30. ábra Létrehozott csoport 43](#_Toc195699478)

[31. ábra Barát gombok 44](#_Toc195699479)

[32. ábra Példa egy barátkérelemre 44](#_Toc195699480)

[33. ábra Beállítások: Fiókom menüpont 45](#_Toc195699481)

[34. ábra Beállítások: Profil menüpont 46](#_Toc195699482)

[35. ábra Bejelentkezések menüpont 47](#_Toc195699483)

[36. ábra Megjelenés menüpont 47](#_Toc195699484)

[37. ábra Beállítások: Hibajelentések menüpont 48](#_Toc195699485)

[38. ábra Példa saját üzenetre kattintásról 50](#_Toc195699486)

[39. ábra Példa Mások üzenetére kattintásról 50](#_Toc195699487)

[40. ábra Példa a további információk panelről 51](#_Toc195699488)

[41. ábra Csoport részletei 52](#_Toc195699489)

[42. ábra Csoporttag kezelése 52](#_Toc195699490)

[43. ábra Csoporthoz adás és kilépés 53](#_Toc195699491)