Budapesti Műszaki Szakképzési Centrum

Verebély László Technikum

5-0613-12-03 Szoftverfejlesztő és tesztelő

ChatJS kommunikációs weboldal

Készítették: Horváth Péter Ákos, Vígh Noel Mihály

Konzulens tanárok: Somogyi Erika, Horváth Attila

Budapest, 2025. április

Nyilatkozat a vizsgaremek eredetiségéről

Alulírott Horváth Péter Ákos, Vígh Noel Mihály a BMSZC Verebély László Technikum 5-0613-12-03 Szoftverfejlesztő és-tesztelő képzésében részt vevő tanulók büntetőjogi felelősség tudatában nyilatkoznak és aláírásukkal igazolják, hogy a RushWave Retreat című vizsgaremek saját, önálló munkánk, és abban betartottuk az iskola által előírt, a vizsgaremek készítésére vonatkozó szabályokat.

Tudomásul veszem, hogy a szakdolgozatban plágiumnak számít:  
•    Szó szerinti idézet közlése idézőjel és hivatkozás nélkül,  
•    tartalmi idézet hivatkozás megjelölése nélkül  
•    más publikált gondolatainak saját gondolatként való feltűntetése.

E nyilatkozat aláírásával tudomásul veszem továbbá, hogy plágium esetén szakdolgozatom visszautasításra kerül.

Budapest, 2025. április 16    \_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Hallgatók aláírásaTartalomjegyzék

Tartalomjegyzék

**Nincsenek tartalomjegyzék-bejegyzések.**

1. Bevezetés

A vizsgraremekünk 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. A webes megoldást több okból választottuk: egyrészt, hogy bármilyen eszközről elérhető legyen az alkalmazás, másrészt, hogy elsajátítsuk a Next.js és Socket.IO technológiákat, amelyek szakmai kihívást, de egyben fejlődési lehetőséget is jelentettek számunkra.

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.

1. Fejlesztői dokumentáció
2. Alkalmazott fejlesztő eszközök

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

1. Alkalmazott technológiák

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

1. 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

1. Fejlesztési ütemterv

|  |  |
| --- | --- |
| Határidő | Feladat |
| 2024. december 19 | Témaválasztás, csoporalakítás |
| 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 |

1. Fejlesztés menete
2. 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.

1. 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 neve, kötelező kitöltésű |
| userAgent | szöveges, a böngésző és operációs rendszer 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 |

themes

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

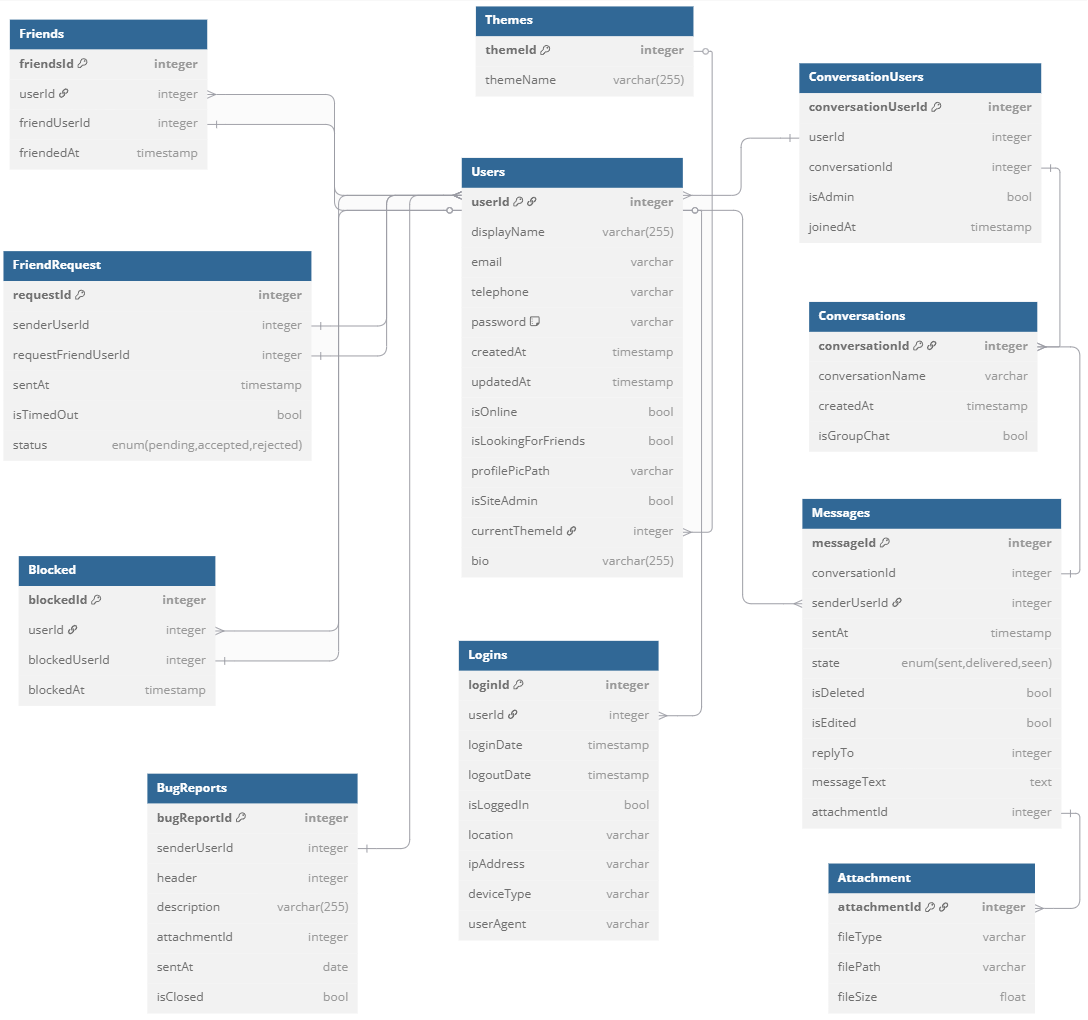
users

|  |  |
| --- | --- |
| 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 |
| 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.

Az adatbiztonság magas szintű biztosítása érdekében a felhasználói jelszavakat olyan titkosítási eljárással tároljuk, amelyet hashelésnek nevezünk. A hashelés során a jelszavakat egy egyirányú kriptográfiai algoritmus segítségével alakítjuk át egy egyedi, rögzített hosszúságú kódolt értékké, amely lehetetlenné teszi, hogy a titkosított jelszót visszaalakítsuk az eredeti jelszóra. Ezáltal még akkor is védelmet biztosítunk, ha valaki hozzáfér az adatbázishoz, hiszen az eredeti jelszó nem tárolódik közvetlenül, csak annak hashelt változata.

Továbbá, a biztonság növelése érdekében minden egyes jelszóhoz úgynevezett „só” értéket generálunk. A só egy véletlenszerű karakterlánc, amelyet a jelszóhoz adunk a hashelési folyamat során, így biztosítva, hogy még ha két felhasználó ugyanazt a jelszót választja is, a titkosított változatok különbözni fognak. Ezáltal a hashelt jelszavak még biztonságosabbá válnak, mivel a só értékek miatt azok nem lesznek könnyen visszafejthetőek olyan támadásokkal szemben, mint amilyen a szivárványtáblák használata. A szivárványtáblák olyan előre számított hashelt jelszavak gyűjteményei, amelyeket a támadók a jelszavak gyors visszafejtésére alkalmazhatnak. A só hozzáadása biztosítja, hogy ez a fajta támadás lényegesen nehezebbé válik, mivel minden jelszó egyedi titkosítást kap.



1. Asztali alkalmazás főbb algoritmusai

Lorem ipsum…

1. Frontend alkalmazás főbb algoritmusai // Webalkalmazás főbb algoritmusai

**

Valós idejű üzenetküldés során a kód a send\_message eseményt figyeli, és ellenőrzi, hogy az üzenethez tartozik-e címzett. Ha van címzett, az üzenetet először a küldő klienshez továbbítja, majd megkísérli azonosítani a címzett socket azonosítóját. Sikeres azonosítás esetén az üzenet közvetlenül eljuttatásra kerül a címzetthez.



A handleSendMessage nevű függvény arra szolgál, hogy amikor egy felhasználó beír egy üzenetet a chatbe, akkor azt elküldje a szervernek, és megjelenítse másoknak is a beszélgetésben. Először elküldi az üzenetet a szervernek, és mellé csomagolja az olyan fontos adatokat, mint például hogy ki küldi, melyik beszélgetéshez tartozik, mi az üzenet szövege, és hogy egy másik üzenetre válaszol-e vele. Ezután megvárja, hogy a szerver visszaküldje a választ, amiben többek között benne van az új üzenet azonosítója és hogy szerkesztve vagy törölve lett-e. Ebből az adathalmazból a függvény létrehoz egy új objektumot, amit aztán elküld a többi felhasználónak is a chat alkalmazáson belül, hogy ők is azonnal láthassák az új üzenetet. Ha minden rendben zajlott, akkor kitörli a szövegmezőből az üzenetet, törli a válasz kijelölését, és visszaállítja a fókuszt az üzenetíró mezőre, hogy lehessen folytatni az írást.



A handleSendMessage függvény először ellenőrzi, hogy az üzenet hossza meghaladja-e a 20 000 karaktert. Ha igen, a rendszer figyelmeztetést küld a felhasználónak, amely arra kéri, hogy rövidítse le az üzenetet. Amennyiben az editMessage értéke igaz, a függvény a szervernek elküldi a módosítani kívánt üzenet azonosítóját és az új szöveget. Ha az editMessage értéke hamis, akkor a függvény az üzenetküldési folyamatot folytatja, amely az aktuális üzenet gyors és zavartalan továbbítását biztosítja a szerverhez és a többi felhasználóhoz.



A handleSendMessage nevű függvény fő célja, hogy amikor egy felhasználó beír egy üzenetet a chatbe, azt továbbítsa a szerverre, majd megjelenítse azt mások számára a beszélgetésen belül. Az üzenet elküldésekor elküldi azokat az alapvető adatokat, mint például a küldő személye, az üzenethez tartozó beszélgetés azonosítója, az üzenet szövege, valamint annak információját, hogy az üzenet egy másik üzenetre válasz-e. A szerver válasza alapján a függvény létrehoz egy új objektumot, amelyet valós időben továbbít a többi felhasználónak, lehetővé téve, hogy az új üzenetet azonnal láthassák. A folyamat végén a függvény törli a szövegmező tartalmát, eltávolítja a válasz kijelölését, és visszaállítja a fókuszt az üzenetíró mezőre, ezzel támogatva az írás folyamatosságát és gördülékenységét.



A deleteMessage nevű függvény arra való, hogy a felhasználó törölni tudja a saját üzeneteit a chatből. Először megnézi, hogy tényleg az törli-e az üzenetet, aki eredetileg írta. Ha nem egyezik a felhasználó az üzenet küldőjével, akkor nem engedi végrehajtani a törlést, és egy hibaüzenetet mutat.

Ha viszont a felhasználó jogosult törölni az üzenetet, akkor kétféleképpen is elvégzi a törlést. Az egyik az, hogy azonnal értesíti a másik felhasználót, hogy ez az üzenet törlésre került – ezt a WebSocket segítségével küldi ki, így a többiek chatje is frissül. A másik dolog, hogy egy üzenetet küld a szervernek, hogy az adatbázisban is jelezze: ez az üzenet törölve lett. A rendszer nem teljesen törli ki az üzenetet, csak beállítja az IsDeleted-et 1-re

Ha a szerver válasza azt mondja, hogy minden rendben ment, akkor a program frissíti a képernyőn is az üzeneteket úgy, hogy a törölt üzenet meg legyen jelölve. Így a felhasználó látja, hogy az adott üzenet már nincs ott.



A handleAddFriend nevű függvény arra szolgál, hogy egy felhasználó barátkérést tudjon küldeni valaki másnak. Először megnézi, hogy a szükséges adatok – vagyis a másik felhasználó neve és azonosítója – meg vannak-e adva. Ha ezek közül valamelyik hiányzik, akkor a függvény nem csinál semmit, és azonnal leáll.

Ha minden adat megvan, akkor a függvény elindítja a küldési folyamatot: küld egy HTTP POST kérést a szerver felé. Ebben benne van, hogy ki küldi a barátkérést, és hogy kinek szól a kérés.

Ha a szerver azt válaszolja, hogy sikeresen fogadta a kérést, akkor a függvény küld egy valós idejű jelzést (WebSocket eseményt) a másik felhasználónak, hogy új barátkérése érkezett. Ezután kitörli a beírt adatokat a mezőkből, és kiír egy üzenetet, hogy a kérés sikeres volt.

A függvény végén minden visszaáll az eredeti állapotába, hogy ne maradjon aktív a betöltési állapot, és a felhasználó tudjon új kérést küldeni. Az egész folyamat úgy van megcsinálva, hogy a felhasználó mindig tudja, mi történik – legyen szó sikeres vagy sikertelen barátkérésről.

1. Backend alkalmazás főbb algoritmusai



Ez az API végpont arra való, hogy egy adott beszélgetés összes üzenetét lekérje. A beszélgetést egy azonosító (conversationId) alapján tudjuk megadni, és a rendszer nemcsak az üzeneteket küldi vissza, hanem azoknak a felhasználóknak az adatait is, akik ezeket írták – például a nevüket és a profilképüket. Így a felhasználók a saját oldalukon szépen meg tudják jeleníteni a beszélgetés teljes történetét.

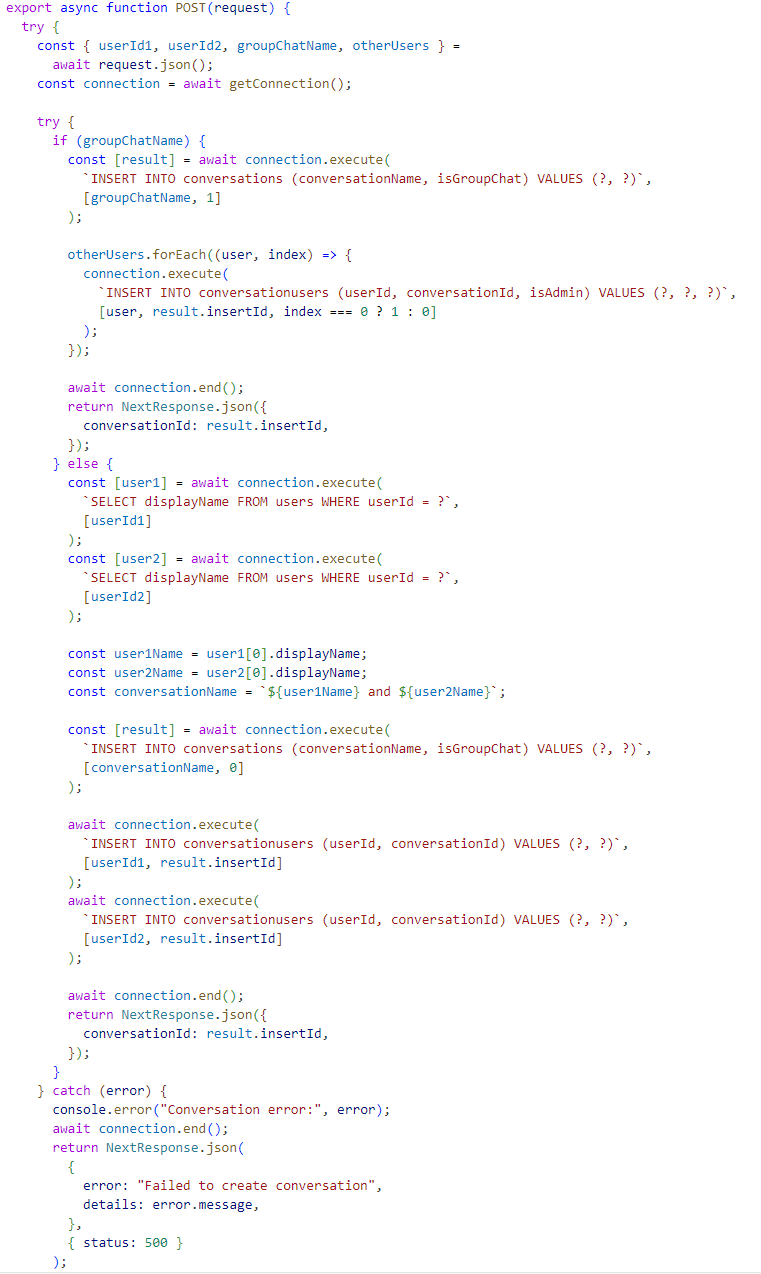
Az egész folyamat úgy kezdődik, hogy a rendszer kiolvassa a beszélgetés azonosítóját az URL-ből, például így: /api/messages?conversationId=123. Ezután létrehozza a kapcsolatot az adatbázissal, majd végrehajt egy lekérdezést, amivel minden olyan üzenetet lekér, ami ehhez a beszélgetéshez tartozik. Az üzenetekhez hozzákapcsolja annak az embernek az adatait is, aki küldte őket.

Fontos, hogy az üzenetek időrendben kerülnek visszaadásra, vagyis a legrégebbitől a legújabbig lesznek sorba rakva. Ha minden jól megy, a válasz egy JSON típusú lista lesz, amely minden egyes üzenethez tartalmazza például a szöveget, mikor lett elküldve, ki küldte, és hogy szerkesztették-e vagy törölték-e. Ha viszont valamilyen hiba történik – például nem jó az azonosító vagy valami gond van az adatbázissal –, akkor a rendszer visszaad egy hibaüzenetet is, hogy mi volt a gond.



Ez a POST típusú API végpont felelős azért, hogy új üzeneteket fogadjon és elmentse őket az adatbázisba. Mielőtt az üzenetet mentené, ellenőrzi, hogy minden szükséges adat – például a beszélgetés és a küldő azonosítója, valamint az üzenet szövege – helyesen szerepel-e, és hogy az üzenet hossza nem haladja-e meg a megengedett határt. Ha valami hiányzik vagy hibás, a rendszer hibaüzenetet küld vissza.

Ha minden rendben van, az üzenetet elmenti, majd visszaküldi a mentett adatokat, például az üzenet szövegét, idejét és a címzett azonosítóját. Emellett hiba esetén naplózza a problémát, így biztosítva, hogy a rendszer stabil és jól karbantartható maradjon. Ez a végpont tehát az üzenetküldés megbízható működését segíti elő az alkalmazásban.



Ez az API route-ban található szkript egy POST-kérést kezel, amelynek célja egy új beszélgetés (chat) létrehozása – lehet privát (két felhasználó között) vagy csoportos. A kérelemből kiolvassa a szükséges adatokat: privát chat esetén két felhasználó azonosítóját, csoportos esetén a csoport nevét és a résztvevők listáját. Ezután létrejön az adatbáziskapcsolat, majd attól függően, hogy csoportos vagy privát beszélgetésről van szó, más-más műveleteket hajt végre. Csoportos chat esetén új rekord jön létre a conversations táblában, ahol az első felhasználó admin jogosultságot kap, a többiek nem, és minden résztvevő bekerül a conversationusers táblába. Privát beszélgetésnél a két felhasználó neve alapján generálódik a beszélgetés neve, és mindketten hozzáadásra kerülnek a conversationusers táblához. Mindkét esetben a kapcsolat lezárul, és visszatér a létrehozott beszélgetés azonosítójával. Hibakezelés során minden esetben biztosított a kapcsolat lezárása, és részletes hibaválasz (500 Internal Server Error) kerül visszaküldésre.

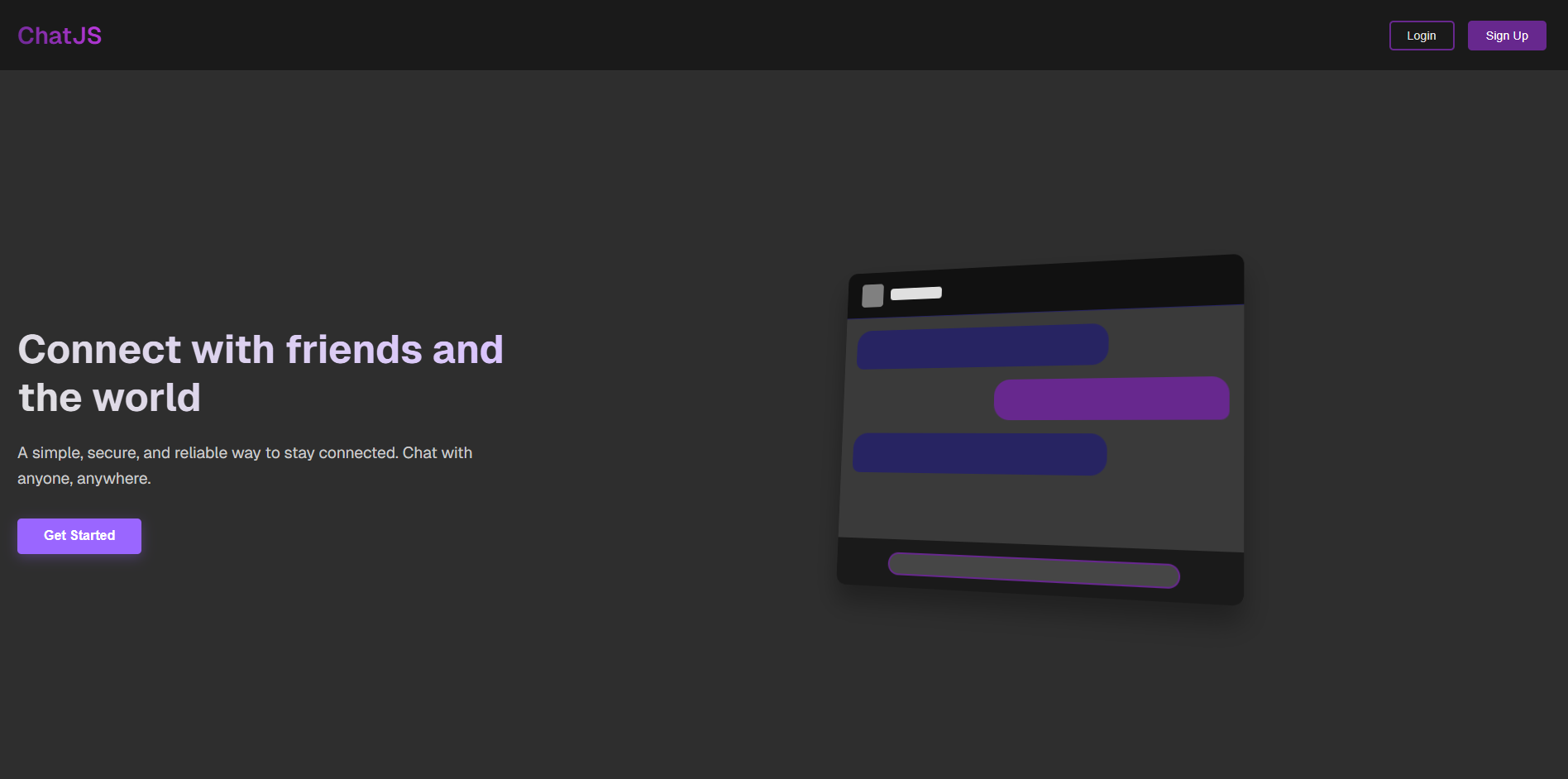
1. Tesztelés

peti will do it yeaaa :DDD

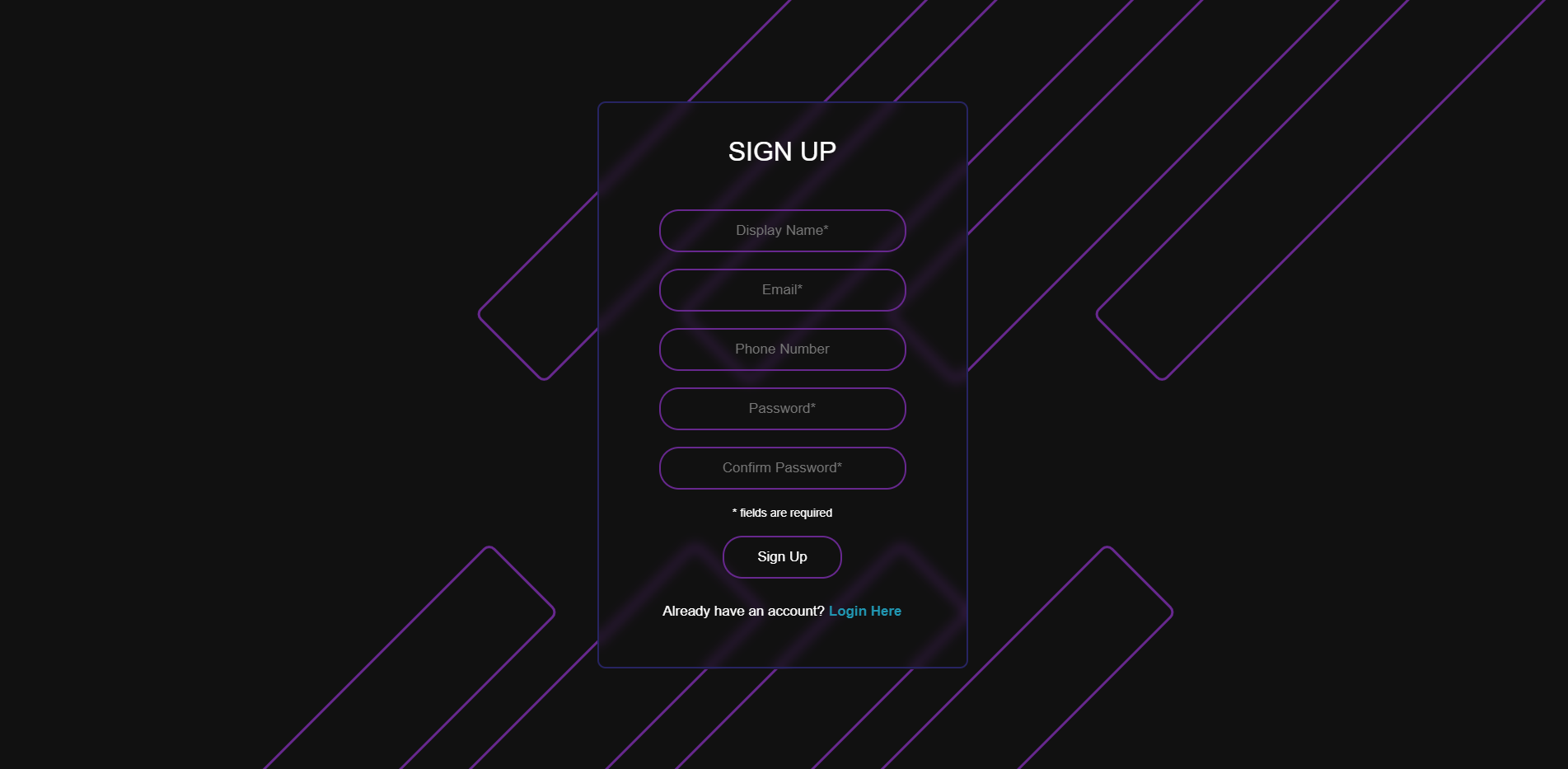
1. Felhasználói dokumentáció
2. ChatJS asztali alkalmazás

Lorem impum

1. ChatJS webalkalmazás

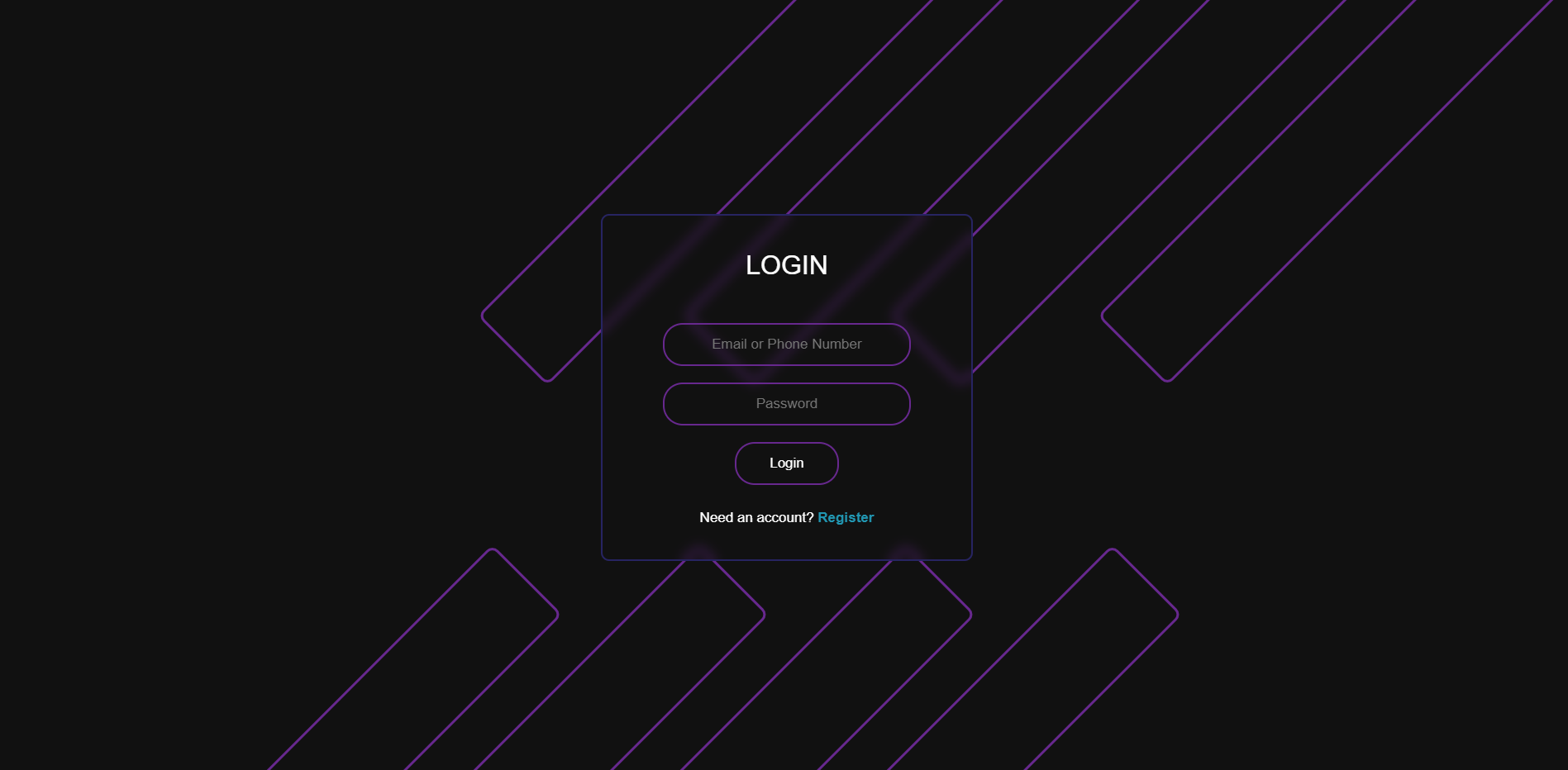


A felhasználó először egy landoló oldalra érkezik, amely az oldal bemutatkozó és tájékoztató felülete. Itt lehetősége van megismerkedni az oldal előnyeivel és funkcióival. Amennyiben a felhasználó úgy dönt, hogy érdeklődik a szolgáltatás iránt, a képernyő közepén elhelyezkedő „Get Started” gomb segítségével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



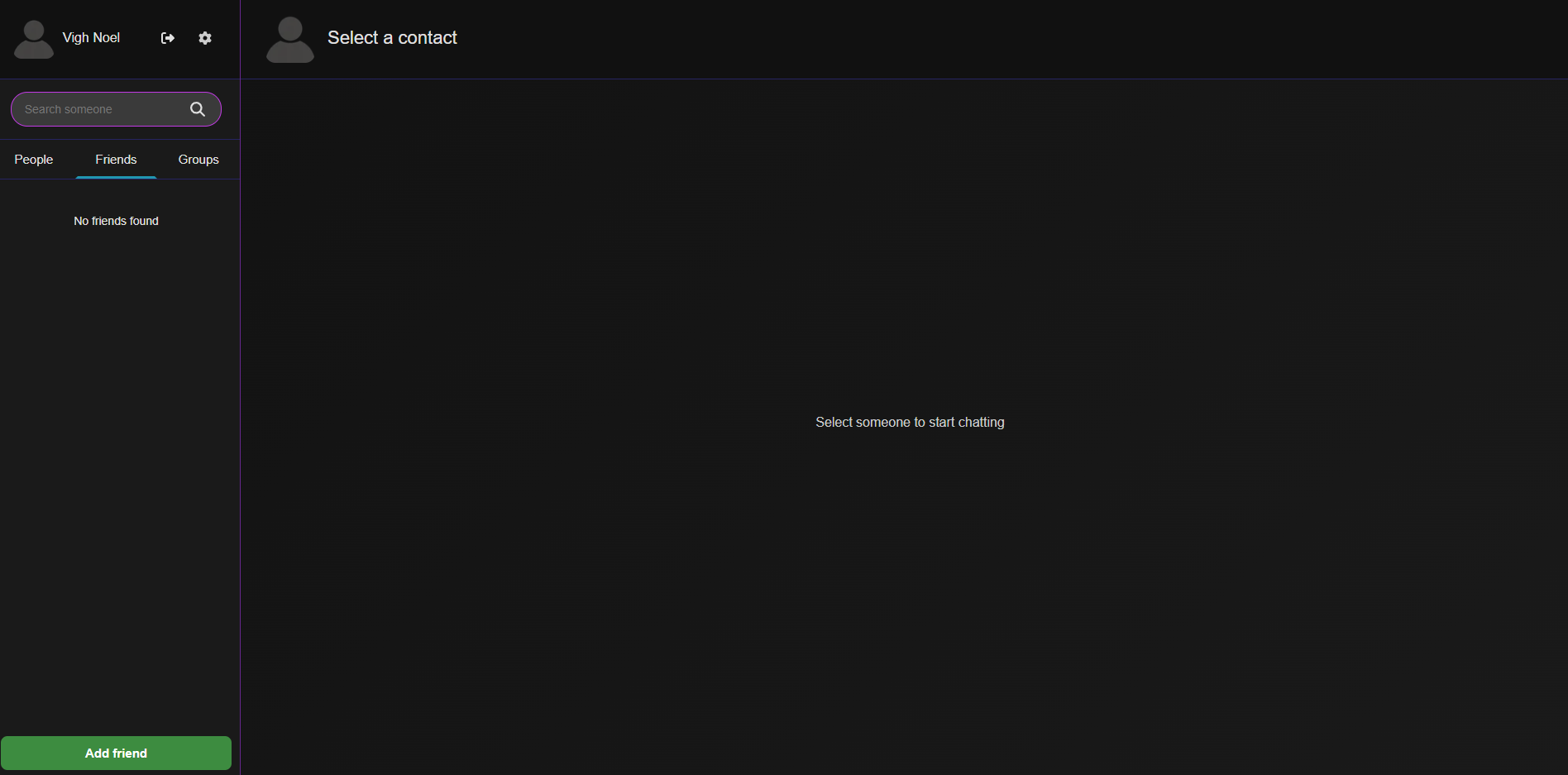
A „Sign up” gomb megnyomása után a felhasználó átirányításra kerül a regisztrációs oldalra, ahol a kötelezően kitöltendő mezők csillaggal (\*) vannak jelölve. A regisztrációs folyamat során a felhasználónak először meg kell adnia a megjelenített nevét (amelyet más felhasználók is látnak az oldalon, és több felhasználó is használhatja ugyanazt a nevet). Ezt követi az e-mail cím megadása, valamint egy jelszó és annak megerősítése. A telefonszám megadása nem kötelező. A kötelező mezők kitöltése után a felhasználó a „Sign up” gombbal véglegesítheti a regisztrációt, amelyet követően át lesz irányítva a bejelentkezési oldalra.

Amennyiben a felhasználó nem megfelelő adatokat adott meg, vagy bármely mezőt kihagyott, a rendszer hibát jelez a felhasználónak, és egy hibauzenetet küld. Abban az esetben, ha a felhasználó már regisztrált a rendszerben, az oldal alján található „Login Here” szövegre kattintva átirányításra kerül a bejelentkezési oldalra.



A bejelentkezési oldalon a felhasználónak meg kell adnia a regisztrált e-mail címét vagy telefonszámát, valamint a jelszavát. Ezt követően a „Login” gombra kattintva sikeresen bejelentkezhet az oldalra. Hibás vagy hiányzó adatok megadása esetén a rendszer szintén hibauzenetet küld a felhasználónak. Amennyiben a felhasználó még nem rendelkezik fiókkal, az oldal alján található „Register” gombra kattintva visszakerülhet a regisztrációs oldalra. A bejelentkezési oldal a landoló oldalról is elérhető a jobb felső sarokban található gombbal.

A sikeres bejelentkezést követően a felhasználó a chatfelületen találja magát. A felhasználói felület a következőképpen van elrendezve:



Bal oldali menü:

A bal oldali menü tetején a felhasználó profilképe és a megadott felhasználói név található.

Ezt követi egy kilépés gomb, amely lehetőséget biztosít arra, hogy a felhasználó elhagyja az oldalt.

Az utolsó elem a beállítások gomb. Rákattintva megnyílik a beállítások fül, ahol három lehetőség közül választhat a felhasználó. Az első lehetőség a „My Account”, ahol a felhasználó a jelszót és az e-mail címet módosíthatja. A második a „Profile”, ahol a felhasználó módosíthatja a megjelenített adatait, mint például a profilképét, a felhasználói nevet és a bemutatkozó szöveget. Az utolsó lehetőség az „Appearance”, ahol az oldal megjelenését, mint például a színvilágot és a betűtípusokat testre szabhatja.

A profil alatt található a keresőmező, ahol alapértelmezetten a barátok között kereshetünk, azonban a „People” vagy „Groups” menüpontokra kattintva a keresést a felhasználók nevei vagy csoportok alapján is végezhetjük.

People menüpont:

A „People” menüpont alatt azok a felhasználók találhatók, akik barátokat keresnek. A felhasználó felveheti őket barátnak, azonban barátság nélkül is lehet velük kommunikálni.

A menü legalján található az „Add Friend” gomb, amely egy új menüt nyit meg. Itt a felhasználó megadhatja egy másik felhasználó nevét vagy ID-ját, és a „Send Request” gombra kattintva barátkérést küldhet.

Ha a felhasználónak aktív barátkérelmei vannak, akkor a „New Requests” gomb megjelenik, amely segítségével láthatóvá válnak az aktív barátkérések. Minden egyes barátkérés mellett két gomb található: „Accept” (elfogadás) és „Reject” (elutasítás).

Elfogadás esetén a barátok fülön megjelenik a felhasználó, ahol a profilkép, a név, és az utolsó üzenet is láthatóvá válik. A státusz zöld körrel jelzi, hogy online, míg a szürke kör azt mutatja, hogy offline.

A felhasználóra jobb kattintva megjelenik egy menü, amelyben a „Block” gomb segítségével letiltható a felhasználó. A letiltott felhasználó a barátlistán marad, de nem tud üzenetet küldeni. A „Delete” gombbal pedig eltávolítható a barátok listájáról.

Csoportok fül:

A „Groups” fülön a felhasználó láthatja azokat a csoportokat, amelyekhez csatlakozott. Ha még nincs csoportja, akkor az oldal alján található „Create a New Group” gombra kattintva új csoportot hozhat létre.

A csoport létrehozásához először meg kell adni a csoport nevét, majd a „Next” gombra kattintva egy új ablakban a felhasználó a barátai közül választhat, hogy kit szeretne hozzáadni a csoporthoz. A „Back” gombbal visszaléphet az előző képernyőhöz, míg a „Create” gombbal létrehozhatja a csoportot.



Középső menü – chat felület:

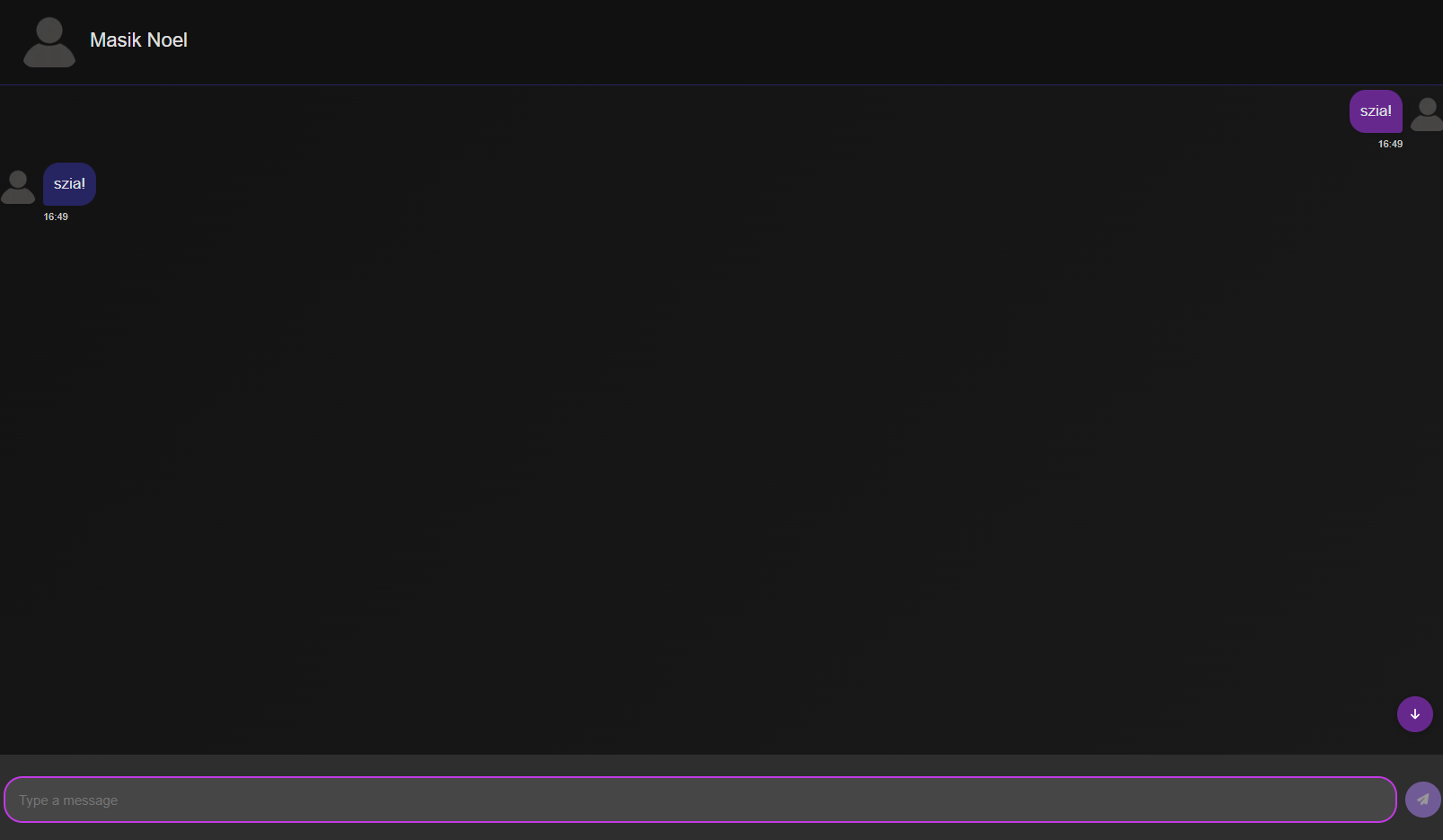
A tetején található a kiválasztott felhasználó profilképe és neve. Ha nincs kiválasztva felhasználó, akkor „Select a Contact” felirat jelenik meg.

Az üzenetek megjelenítése alatt található a chat ablak, amelyben a felhasználó által küldött üzenetek és a másik fél válaszai jelennek meg. A felhasználó üzenetei lila színnel, míg a másik fél üzenetei kék színnel jelennek meg. Az üzenet alatt az üzenet elküldésének időpontja látható.

Minden üzenetre jobb kattintva egy menü jelenik meg, amely az alábbi műveleteket teszi lehetővé: „Reply To” (válaszolás), „Copy Text” (szöveg másolása), „Edit Message” (üzenet módosítása), és „Delete” (üzenet törlése). Ha az üzenetet módosítják, akkor az elküldés ideje mellett „Edited” (módosítva) felirat jelenik meg.

Alul található input mező:

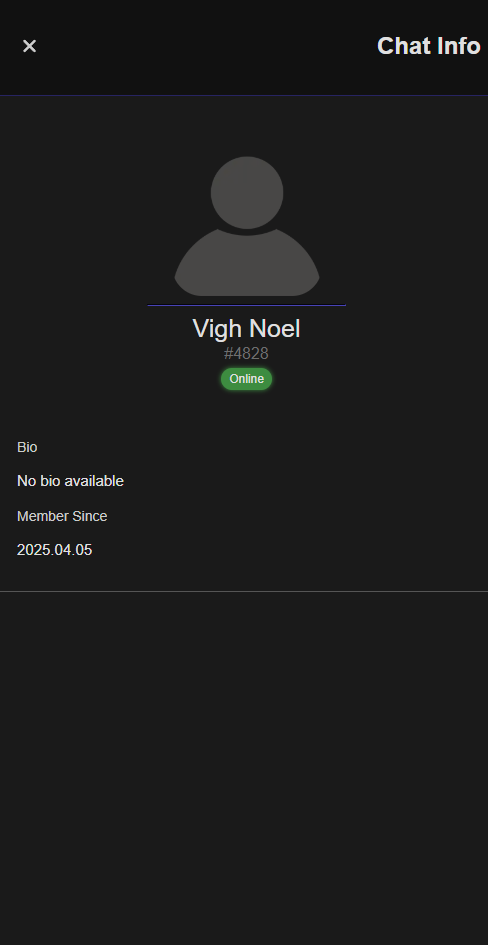
Az üzenetek írása az alul található szövegdobozba történik, és az üzenet a „Enter” gomb megnyomásával vagy a melletti gombbal küldhető el.



Jobb oldali menü:

A jobb oldali menüben a kiválasztott felhasználó részletes profilja jeleníthető meg, beleértve a profilképet, nevet, ID-t, státuszt, bemutatkozó szöveget, és a csatlakozás dátumát.

A menü bal felső sarkában található „X” gomb segítségével a jobb oldali menü bezárható.



1. 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.
* Képek és 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.

1. 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.

Dizájn szempontjából minimalista és funkcionális megközelítést alkalmaztunk, kizárólag HTML és CSS segítségével, külső keretrendszerek nélkül. Bár elégedettek vagyunk az eredménnyel, a jövőben további fejlesztésekre van lehetőség, hogy a platform funkcionalitását és stabilitását tovább növeljük.

1. Ábrajegyzék

Mellékletek