# Chatex

# Dokumentáció

# [*Noszlopy Gáspár Közgazdasági Technikum*](http://www.ngkszki.hu/)

# Készítette: Szép Dániel,

# Kiss Levente,

# Szabó Richárd

# Tartalomjegyzék

Tartalomjegyzék

Bevezetés

2.1 A feladat címe

2.2 A feladat rövid ismertetése

Elvárások a feladattal kapcsolatban

3.1 Környezet

3.2 Felhasználandó programozási nyelv

3.3 Megoldás Formátuma

3.4 Modulok

Specifikáció

4.1 Megjelenés

4.2 Funkciók

4.3 Titkosítás

Dokumentáció

5.1 Erőforrásterv, munkaidő-nyilvántartás

5.2 Forráskód dokumentáció

5.3 Felhasználói dokumentáció

A projekt értékelése

6.1 A feladat értékelésének felhasználói oldali szempontjai

6.2 A feladat értékelésének technikai szempontjai

Projekt adatlap

# 2 - Bevezetés

## 2.1 A témaválasztás megindoklása

## Alkalmazásunk a Messenger jelenlegi (v485.2.0.68.111) hibáit javítja ki, köztük a végpontok közötti titkosítástól való teljesítmény problémákat (a felhasználó dönthet a bekapcsolásáról) és a kezelő felületek inkonzisztens megjelenését egy egységes felülettel.

## 2.2 Célkitűzés

A Chatex alkalmazás célja, hogy egy alternatívát nyújtson a híres Messenger helyett, mégpedig úgy, hogy csak az egymás közötti csevegésre fókuszál minden olyan funkció nélkül, ami nem ezt a célt szolgálja. Más szóval, a Chatex használata egy sokkal könnyebb, gyorsabb, és felhasználó barátibb környezetet nyújt, míg ugyanúgy megtartja a játékos funkciókat. **A fejlesztés közben be kell tartani a Clean Code alapszabályait.**

### Főbb jellemzők:

1. Villámgyors üzenetküldés késleltetés nélkül
2. Biztonságos adatkezelés és titkosítás a felhasználói adatok védelmében
3. Modern, letisztult felület, amely könnyen kezelhető
4. Innovatív funkciók, amelyekkel az élmény még jobb, mint a hagyományos alkalmazásokban

**3** - **Fejlesztői környezet**

A chat applikációnk elkészítéséhez többféle fejlesztőeszközöket használtunk. A kód nagy részét Visual Studio Code-ban írtuk meg.  
Az XAMPP program használatával hajtottuk végre a MySQL adatbáziskezelést és a PHP-t, főleg hogy lehetővé teszi, hogy lokálisan fejlesszük és teszteljük applikációnkat. Az adatbázisunk karbantartására a phpMyAdmin segítségét vettük igénybe, utána a MySQL adatbáziskezelő rendszer tárolja és kezeli a felhasználókhoz és a chatekhez tartozó adatokat

# 3 - Elvárások a feladattal kapcsoltban

## - Operációs rendszer, környezet

* Android

## 3.2 - Felhasználandó programozási nyelv

* Flutter, Dart, PHP(Backend)

## 3.3 - Megoldás formátuma

* Forráskód állományok
* Teljes projekt környezet
* Forráskód dokumentáció
* Erőforrás terv és munkaidő nyilvántartás

1. **- Github környezet**

A vizsgaremekhez kiírt feltételekhez és munkánk megkönnyítéséhez alakítottunk ki egy Github környezetet applikációnkhoz.

egy közös email cím létrehozásával végeztük el a környezet elkészítését.

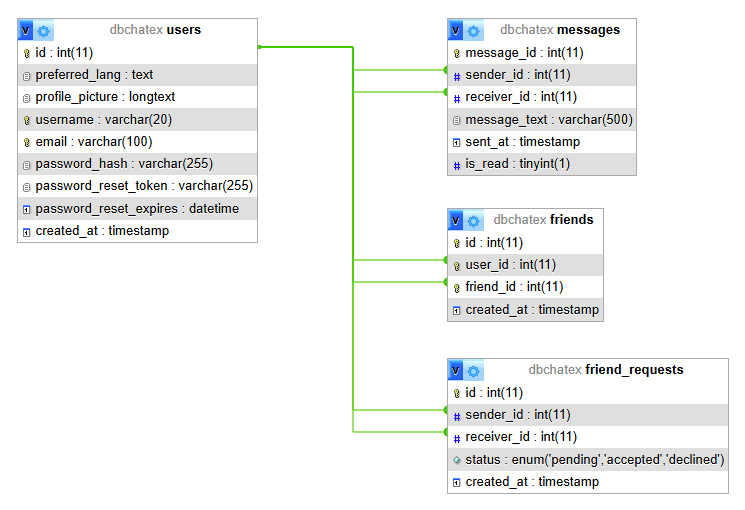
A közös emaillel csináltunk egy repository-t Messengeres-vizsgaremek néven és ezután hozzáadtuk egymás fiókjait, hogy mindegyikőnk fejlesztése könnyen nyomkövethető és zökkenőmentes legyen.

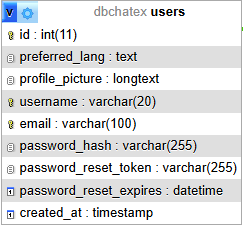
Az elkészült vizsgaremek tartalmazza az adatbázist, programkódot, dokumentációt és az előadást, amit a következő Github repositoryban érhető el: [Chatex vizsgaremek](https://github.com/smolelevent/Messengeres-vizsgaremek). itt található az összes szükséges anyag a projekt teljes megértéséhez.

**5 – A kialakított adatszerkezet részletes bemutatása**

Chat applikációnk táblái strukturáltak, szervezettek és hatékonyan kezelik, tárolják az adatokat.

Az adatbázis táblái

A Users tábla

id (fő azonosító): Ez a mező az adott felhasználó azonosítója. Ez az elsődleges kulcs, szóval egyedi minden egyes felhasználónak ezért segít megkülönböztetni különböző felhasználókat a táblában. Ehhez az azonosítóhoz kapcsolódik az összes többi azonosító, mivel ez a fő azonosító.

preferred\_lang (kiválasztott nyelv): Ez a mező a felhasználó által kiválasztott nyelv adatait tárolja, Egyelőre még csak magyar és angol nyelv elérhető.

profile\_picture (profilkép): Ez a mező az adott felhasználó profilképe, a felhasználó állítja be egy JPEG, PNG, GIF, Animated GIF, WebP, Animated WebP, BMP, and WBMP formátumu képpel, ha nem állít be profilképet, az applikáció az alapértelmezettet adja a felhasználónak.

username (felhasználónév): Ez a mező tartalmazza az adott felhasználó által megadott felhasználónevet, ezzel a névvel tudnak a többi felhasználók rákeresni erre a személyre, ez a név jelenik meg a chatekben.

email (email 😊): Ez a mező tartalmazza az adott felhasználó által megadott email címet, erre az email címre fogja megkapni a felhasználó az összes üzenetet a rendszertől (pl.: új jelszó kérés).

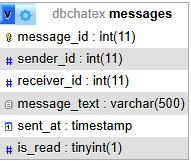
password\_hash (jelszó): Ez a mező tartalmazza az adott felhasználó által megadott jelszót, ezzel tud csak a felhasználó bejelentkezni fiókjába, és he elfelejti, új jelszót kell kérnie mert a Chatex nem fogja tudni megadni, mert az adatbázisban csak egy hash code-ként van elmentve.

password\_reset\_token (jelszó helyreállítása): Ez a mező tartalmazza az adott felhasználónak éppen függőbe lévő jelszó helyreállítási kérését, ha nincs, akkor üres, ha van, akkor az adatbázis mutatja a tokent.

password\_reset\_expires (jelszó helyreállításának élettartama): Ez a mező tartalmazza az adott felhasználónak a jelszó helyreállításának élettartamát (15 perc), ha a felhasználó nem végezte el a dolgát ebben az időtartamban, a jelszó helyreállításának tokenje automatikusan törölve lesz az adatbázisból és új kérést kell nyitnia a felhasználónak.

created\_at (fiók készült ekkor): Ez a mező tartalmazza azt, hogy adott felhasználó pontosan mikor regisztrálta fiókját.

A messages tábla

message id (üzenet azonosító): Ez a mező tartalmazza minden egyes üzenetnek az azonosítóját, hogy miután a felhasználó kilép az applikációból, ne tünjenek el az üzenet mikor visszalép, ez a táblának az elsődleges kulcsa, a mező INT típusú és maximum 11 karaktert fogad be.

sender\_id (adó azonosító): Ez a mező tartalmazza az üzenetet küldő felhasználónak az azonosítóját, ezzel biztosítjuk, hogy az üzenet biztos ez a kettő személy között lesz elmentve, a mező INT típusú és maximum 11 karaktert fogad be.

receiver\_id (vevő azonosító): Ez a mező tartalmazza az üzenetet vevő felhasználónak az azonosítóját, ezzel biztosítjuk, hogy az üzenet biztos ez a kettő személy között lesz elmentve, a mező INT típusú és maximum 11 karaktert fogad be.

message\_text (üzenet): Ez a mező tartalmazza az üzenetek tartalmát, amihez hozzá van adva az üzenet azonosító, ezzel a kettővel nem fog elveszni se maga az üzenet, se az üzenet tartalma, a mező VARCHAR típusú és maximum 500 karaktert fogad be.

sent\_at (elküldve ekkor): Ez a mező tartalmazza azt az időpontot, amikor az adó felhasználó által elküldött üzenetet a vevő felhasználó megkapta, a mező TIMESTAMP típusú.

is\_read (láttam): Ez a mező tartalmazza azt, hogy az elküldött üzenetet a vevő felhasználó látta-e vagy nem, a mező TINYINIT típusú és csak egyetlen egy karaktert fogad be.

A friends tábla

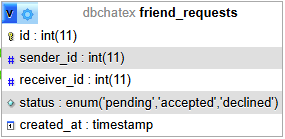
id (azonosító): Ez a mező tartalmazza a barátlistán lévő felhasználók azonosítóját, ez az azonosító az elsődleges kulcs és INT típusú, maximum 11 karaktert fogad be.

user\_id (felhasználó azonosító): Ez a mező tartalmazza a felhasználónak az azonosítóját, ez az egyik rész a barát azonosító megalkozásához, a mező INT típusú és maximum 11 karaktert fogad be.

friend\_id(barát azonosító): Ez a mező tartalmazza a két felhasználó azonosítóját, amit kombinálva megalkotja a barát azonosítót (pl.:1-27), a mező INT típusú és maximum 11 karaktert fogad be

created\_at (barátlistához hozzáadva ekkor): Ez a mező tartalmazza azt az időpontot, amikor a két felhasználó hozzá adta egymást a barátlistájukhoz, a mező TIMESTAMP típusú.

A friend requests tábla

id (azonosító): Ez a mező tartalmazza maga a barátkérelemnek az azonosítóját, hogy az adatbázis feljegyezze a két felhasználó között lehetséges több barátkérelmet is, a mező INT típusú és maximum 11 karaktert fogad be.

sender\_id (barátkérelmet küldő azonosító): Ez a mező tartalmazza a barátkérelmet elküldő felhasználónak az azonosítóját, a mező INT típusú és maximum 11 karaktert fogad be.

receiver\_id (barátkérelmet kapó azonosító): Ez a mező tartalmazza annak a felhasználónak az azonosítóját, aki a barátkérelmet kapta egy másik felhasználótól aki még nincs a barátlistáján, a mező INT típusú és maximum 11 karaktert fogad be.

status (barátkérelem állapota): Ez a mező tartalmazza a barátkérelem állapotát, egyből az elküldés után a „pending” (függőben) attribútumot kapja, ha a vevő felhasználó elutasítja akkor a „declined” (elutasítva) attribútumot kapja, ha elfogadja akkor meg „accepted” (elfogadva) attribútumot kapja, a mező ENUM típusú, és csak „pending”, „accepted” és „decline” -al térhet vissza.

created at (barátkérelem elküldésének időpontja): Ez a mező tartalmazza azt az időpontot, amikor a barátkérelmet elküldték a felhasználónak.

Frontend és a Backend megvalósítása

blablablablaba

Felhasználó regisztrálása

<?php

header("Content-Type: application/json; charset=UTF-8");

header("Access-Control-Allow-Origin: \*");

header("Access-Control-Allow-Methods: POST");

header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");

require\_once \_\_DIR\_\_ . "/../db.php";

//formátum stb. ellenőrzés mert ha pl.: szimuláljuk az adatokat tudjuk hol a hiba, jelszó visszaállító email küldése!

function normalizeEmail($email)

{

    // Gmail esetén: a pontokat és a + utáni részt eltávolítjuk

    if (strpos($email, '@gmail.com') !== false) {

        $emailParts = explode('@', $email);

        $localPart = str\_replace('.', '', $emailParts[0]); // Pontok eltávolítása

        $localPart = explode('+', $localPart)[0]; // + utáni rész eltávolítása

        return $localPart . '@gmail.com';

    }

    return $email; // Más e-mail szolgáltatóknál nincs változás

}

$userData = json\_decode(file\_get\_contents("php://input"), true);

// Adatok kinyerése

$username = trim($userData['username']);

$email = normalizeEmail(trim($userData['email']));

$password = trim($userData['password']);

$password\_hash = password\_hash($password, PASSWORD\_DEFAULT);

$language = $userData['language'];

// Ellenőrizzük, hogy az e-mail formátuma helyes-e

if (!filter\_var($email, FILTER\_VALIDATE\_EMAIL)) {

    http\_response\_code(400); // Hibás kérés

    echo json\_encode(["message" => "Érvénytelen email cím!"]);

    exit();

}

// Ellenőrizzük, hogy az email cím már létezik-e az adatbázisban

$checkStmt = $conn->prepare("SELECT id FROM users WHERE email = ?");

$checkStmt->bind\_param("s", $email);

$checkStmt->execute();

$checkStmt->store\_result();

if ($checkStmt->num\_rows > 0) {

    http\_response\_code(409); // Konfliktus

    echo json\_encode(["message" => "Ezzel az emailel már létezik felhasználó!"]);

    exit();

}

// Felhasználó beszúrása az adatbázisba stmt = STATEMENT (állítás)

$stmt = $conn->prepare("INSERT INTO users (preferred\_lang, username, email, password\_hash, created\_at) VALUES (?, ?, ?, ?, NOW())");

if ($stmt === false) {

    http\_response\_code(500); // Belső szerverhiba

    echo json\_encode(["message" => "SQL előkészítési hiba."]);

    exit();

}

$stmt->bind\_param("ssss", $language, $username, $email, $password\_hash);

// Lekérdezés végrehajtása és válasz küldése

if ($stmt->execute()) {

    http\_response\_code(201); // Erőforrás létrehozva

    echo json\_encode([

        "message" => "Sikeres regisztráció!",

        "username" => $username,

        "email" => $email,

        "password\_hash" => $password\_hash,

"preferred\_lang" => $language

]);

} else {

http\_response\_code(500); // Belső szerverhiba

echo json\_encode(["message" => "Sikertelen regisztráció!"]);

}

$stmt->close();

$conn->close();

Ez a PHP szkript egy API végpontot biztosít a felhasználók regisztrációjához. Az API JSON formátumban fogadja a bemenetet, ellenőrzi az adatokat, normalizálja a Gmail-es e-mail címeket, és elmenti az adatokat egy MySQL adatbázisba.

A header beállításoknál előszőr beállítjuk, hogy A válasz JSON formátumban érkezzen, bárhonnan engedélyezett a hozzáférés, Az API kizárólag **POST** metódussal hívható és az API-hoz szükséges fejléceket engedélyezi.

A normalizeEmail funkció használatával a különböző használatú emaileket egy formátummá teszi, így nem kell tökölni az összes lehetséges emailt lecsekkolni.

$userData = json\_decode(file\_get\_contents("php://input"), true);

Átveszi az adatokat és be is olvassa a tárolójukba:

$username = trim($userData['username']);

$email = normalizeEmail(trim($userData['email']));

$password = trim($userData['password']);

$password\_hash = password\_hash($password, PASSWORD\_DEFAULT);

$language = $userData['language'];

Utána ellenőrzi az email formátumát, és ha helyes, akkor tovább megy és ellenőrzi, hogy létezik-e már az email az adatbázisban, utána beszúrja az adatbázisba, ha minden jól megy és kész a regisztráció.

Felhasználó bejelentkezés

<?php

header("Content-Type: application/json; charset=UTF-8");

header("Access-Control-Allow-Origin: \*");

header("Access-Control-Allow-Methods: POST");

header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");

require\_once \_\_DIR\_\_ . "/../db.php";

require\_once \_\_DIR\_\_ . '/../vendor/autoload.php';

use Firebase\JWT\JWT;

$userData = json\_decode(file\_get\_contents("php://input"), true);

$email = trim($userData['email']);

$password = trim($userData['password']);

// Lekérdezzük a felhasználót az email alapján

$stmt = $conn->prepare("SELECT id, preferred\_lang, profile\_picture, username, email, password\_hash FROM users WHERE email = ?");

$stmt->bind\_param("s", $email);

$stmt->execute();

$result = $stmt->get\_result();

$user = $result->fetch\_assoc();

if (!$user || !password\_verify($password, $user["password\_hash"])) {

    http\_response\_code(401); // 401 = Unauthorized

    echo json\_encode(["message" => "Hibás email vagy jelszó!"]);

    exit();

}

// JWT token létrehozása - hitelesítés amivel pl: nem kell újra meg újra bejelentkezni

$issued\_at = time();

$expiration\_time = $issued\_at + (60 \* 60 \* 24); // 24 óra

$payload = [

    "iat" => $issued\_at,

    "exp" => $expiration\_time,

    "sub" => $user["id"],          // Felhasználó ID-ja az adatbázisból

    "username" => $user["username"], // Adatbázisból lekért username

    "email" => $user["email"],

    "preferred\_lang" => $user["preferred\_lang"]

];

$secret\_key = "chatex";

$jwt = JWT::encode($payload, $secret\_key, 'HS256');

http\_response\_code(200); // 200 = OK

echo json\_encode([

    "message" => "Sikeres bejelentkezés",

    "success" => true,

    "token" => $jwt,

    "id" => $user["id"],

    "username" => $user["username"],

    "preferred\_lang" => $user["preferred\_lang"],

    "email" => $user["email"],

    "password\_hash" => $user["password\_hash"],

    "profile\_picture" => trim($user["profile\_picture"], "'\"")

]);

$stmt->close();

$conn->close();

Ez a PHP szkript egy bejelentkezési API végpontot valósít meg JSON Web Token (JWT) használatával. A felhasználó e-mail címe és jelszava alapján hitelesíti a bejelentkezést, és sikeres azonosítás esetén egy JWT tokent generál.

Adatok fogadása: A bejövő JSON kérés tartalmát a **php://input** segítségével olvassuk be és dekódoljuk.  
Adatok ellenőrzése: Az e-mail cím és a jelszó tisztítása (trim()) után az adatbázisban ellenőrizzük a felhasználót.  
Felhasználó keresése: SQL lekérdezéssel próbáljuk megkeresni az e-mail címhez tartozó felhasználót.  
Jelszó ellenőrzése: Ha a felhasználó létezik, a password\_verify() függvénnyel ellenőrizzük a jelszót.  
JWT token generálása: Sikeres bejelentkezés esetén egy 24 órán keresztül érvényes JWT tokent hozunk létre.  
Válasz küldése: A válasz tartalmazza a JWT tokent és a felhasználó adatait.  
Hibakezelés: Hibás bejelentkezési adatok esetén 401-es státuszkódot küldünk vissza.

Jelszó helyreállítás

<?php

header("Content-Type: application/json");

header("Access-Control-Allow-Origin: \*");

header("Access-Control-Allow-Methods: POST");

header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");

use PHPMailer\PHPMailer\PHPMailer;

use PHPMailer\PHPMailer\Exception;

require\_once \_\_DIR\_\_ . "/../db.php"; // Adatbázis kapcsolat

require\_once \_\_DIR\_\_ . '/../vendor/autoload.php'; // PHPMailer betöltése

// JSON adatok beolvasása

$data = json\_decode(file\_get\_contents("php://input"), true);

if (!isset($data["email"])) {

    echo json\_encode(["success" => false, "message" => "Email megadása kötelező."]);

    exit();

}

$email = $data["email"];

// Megnézzük, hogy létezik-e az e-mail az adatbázisban

$stmt = $conn->prepare("SELECT id FROM users WHERE email = ?");

$stmt->bind\_param("s", $email);

$stmt->execute();

$result = $stmt->get\_result();

if ($result->num\_rows > 0) {

    $row = $result->fetch\_assoc();

    $userId = $row["id"];

    // Token generálása és lejárati idő beállítása

    $token = bin2hex(random\_bytes(20));

    $expires = date("Y-m-d H:i:s", strtotime("+15 minutes"));

    // Token mentése az adatbázisba

    $stmt = $conn->prepare("UPDATE users SET password\_reset\_token = ?, password\_reset\_expires = ? WHERE id = ?");

    $stmt->bind\_param("ssi", $token, $expires, $userId);

    $stmt->execute();

    // Jelszó visszaállító link

    $resetLink = "http://localhost/ChatexProject/chatex\_phps/reset\_password/open\_reset\_window.php?token=$token";

    // \*\*📧 PHPMailer konfigurálása és email küldés\*\*

    $mail = new PHPMailer(true);

    try {

        // SMTP beállítások

        $mail->isSMTP();

        $mail->Host = 'smtp.gmail.com'; // SMTP szerver (pl. Gmail)

        $mail->SMTPAuth = true;

        $mail->Username = 'chatexfejlesztok@gmail.com'; // SMTP e-mail címed

        $mail->Password = 'uvatzwfcrjlcujrs'; // SMTP jelszó vagy App Password

        $mail->SMTPSecure = PHPMailer::ENCRYPTION\_STARTTLS;

        $mail->Port = 587;

        $mail->CharSet = 'UTF-8';

        $mail->Encoding = 'base64';

        // Feladó és címzett

        $mail->setFrom('chatexfejlesztok@gmail.com', 'Chatex support (no reply)');

        $mail->addAddress($email);

        // E-mail tartalma

        $mail->isHTML(true);

        $mail->Subject = "Jelszó visszaállítás";

        $mail->Body = "<h1>Kattints az alábbi linkre a jelszó visszaállításához:</h1>

                       <p><a href='$resetLink' target='\_blank'>$resetLink</a></p>

                       <h2>Ez a link 15 percig érvényes.</h2>

                       <p>Chatex</p>";

        // E-mail küldés

        if ($mail->send()) {

            echo json\_encode(["success" => true, "message" => "Helyreállító e-mail elküldve."]);

        } else {

            echo json\_encode(["success" => false, "message" => "E-mail küldése sikertelen."]);

        }

    } catch (Exception $e) {

        echo json\_encode(["success" => false, "message" => "E-mail hiba: {$mail->ErrorInfo}"]);

    }

} else {

    echo json\_encode(["success" => false, "message" => "Nincs ilyen email című felhasználó!"]);

}

$stmt->close();

$conn->close();

Ez egy PHP alapú rendszer, amely segít a felhasználóknak visszaállítani az elfelejtett jelszavukat. Ha valaki elfelejti a jelszavát, megadhatja az e-mail címét, és kap egy linket, amelyen keresztül új jelszót állíthat be.

A script fogad egy HTTP POST kérést, amely tartalmazza a felhasználó e-mail címét.  
Ellenőrzi, hogy az e-mail szerepel-e az adatbázisban.  
Ha igen, akkor generál egy véletlenszerű **token**-t, amelyet eltárol az adatbázisban egy lejárati idővel együtt.  
Létrehoz egy jelszó visszaállító linket, amely ezt a tokent használja.  
Az e-mailt elküldi a felhasználónak PHPMailer segítségével.  
Ha az e-mail küldése sikeres, visszatér egy JSON válasszal.

Ehhez tartozik még a FORM:

<?php

require\_once \_\_DIR\_\_ . "/../db.php";

if ($\_SERVER["REQUEST\_METHOD"] === "POST") {

    header("Content-Type: application/json");

    header("Access-Control-Allow-Origin: \*");

    header("Access-Control-Allow-Methods: POST");

    header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");

    $token = $\_POST["token"] ?? '';

    $newPassword = $\_POST["new\_password"] ?? '';

    if (!$token || !$newPassword) {

        echo json\_encode(["success" => false, "message" => "Érvénytelen kérés."]);

        exit;

    }

    $stmt = $conn->prepare("SELECT id, email FROM users WHERE password\_reset\_token = ? AND password\_reset\_expires > NOW()");

    $stmt->bind\_param("s", $token);

    $stmt->execute();

    $result = $stmt->get\_result();

    if ($result->num\_rows > 0) {

        $row = $result->fetch\_assoc();

        $userId = $row["id"];

        $userEmail = $row["email"];

        $hashedPassword = password\_hash($newPassword, PASSWORD\_BCRYPT);

        $stmt = $conn->prepare("UPDATE users SET password\_hash = ?, password\_reset\_token = NULL, password\_reset\_expires = NULL WHERE id = ?");

        $stmt->bind\_param("si", $hashedPassword, $userId);

        $stmt->execute();

        echo json\_encode(["success" => true, "message" => "Jelszó sikeresen frissítve!"]);

    } else {

        echo json\_encode(["success" => false, "message" => "A jelszó helyreállító email lejárt!"]); //Érvénytelen vagy lejárt token.

    }

    $stmt->close();

    $conn->close();

    exit;

}

// Ha nem POST kérés, hanem GET (az oldal megjelenítése)

$token = $\_GET["token"] ?? '';

if (!$token) {

    die("Érvénytelen token.");

}

// Lekérjük az email címet a token alapján

$stmt = $conn->prepare("SELECT email FROM users WHERE password\_reset\_token = ? AND password\_reset\_expires > NOW()");

$stmt->bind\_param("s", $token);

$stmt->execute();

$result = $stmt->get\_result();

$userEmail = "";

if ($result->num\_rows > 0) {

    $row = $result->fetch\_assoc();

    $userEmail = $row["email"];

}

$stmt->close();

$conn->close();

Ez a PHP program egy jelszó-visszaállító rendszert valósít meg. Ha valaki elfelejti a jelszavát, ezzel a rendszerrel új jelszót állíthat be. Kétféle módon működik:

POST kérés: Ha a felhasználó már kapott egy visszaállító e-mailt és beírja az új jelszavát.  
GET kérés: Ha valaki rákattint a jelszó-visszaállító linkre, megnyílik az oldal, ahol beírhatja az új jelszót.

A rendszer ellenőrzi, hogy a felhasználó valóban jogosult-e a jelszó megváltoztatására.

## 1. Ha a felhasználó elküldi az új jelszót (POST kérés)

A program ellenőrzi, hogy mindkét adat meg van-e adva.  
Megkeresi az adatbázisban, hogy a token érvényes-e.  
Ha a token rendben van:

* + Az új jelszót titkosítva menti el.
  + A token törlésre kerül, hogy ne lehessen újra felhasználni.
  + Sikeres választ küld vissza.

Ha a token hibás vagy lejárt, hibaüzenetet küld vissza.

## 2. Ha a felhasználó megnyitja a jelszó-visszaállító oldalt (GET kérés)

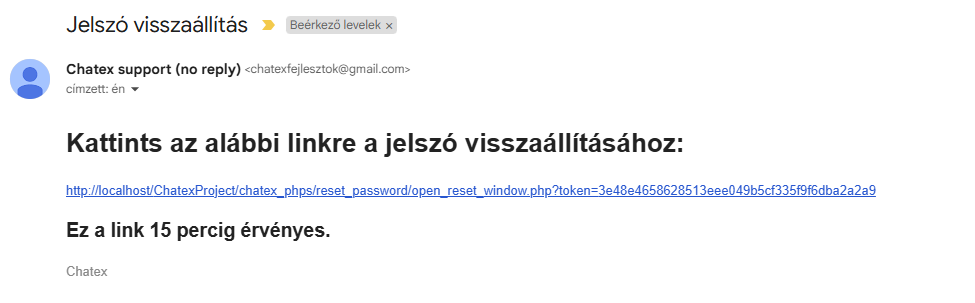
A program ellenőrzi, hogy van-e token az URL-ben.  
Ha igen, megkeresi az adatbázisban az e-mail címhez tartozó adatokat.  
Ha a token érvényes:

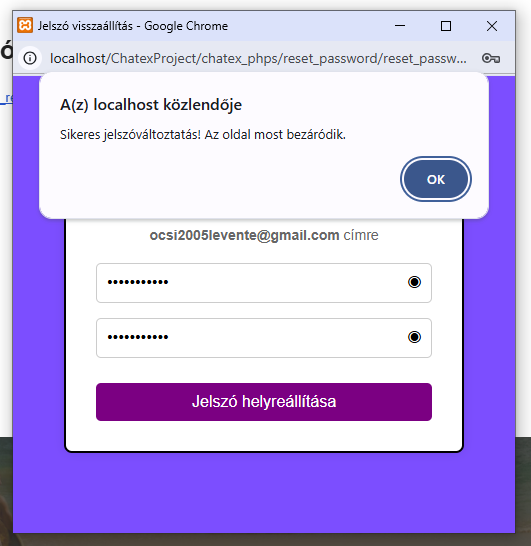
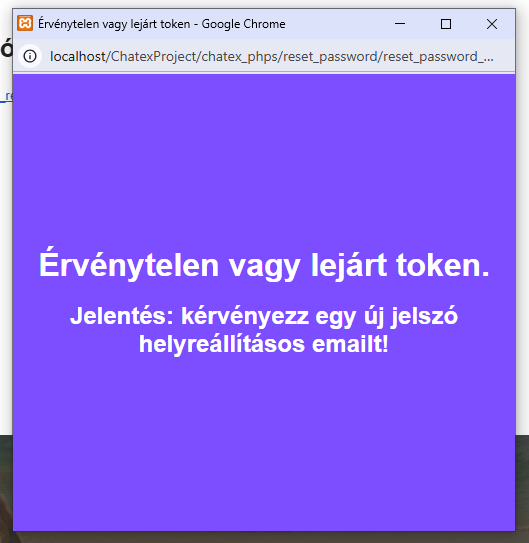
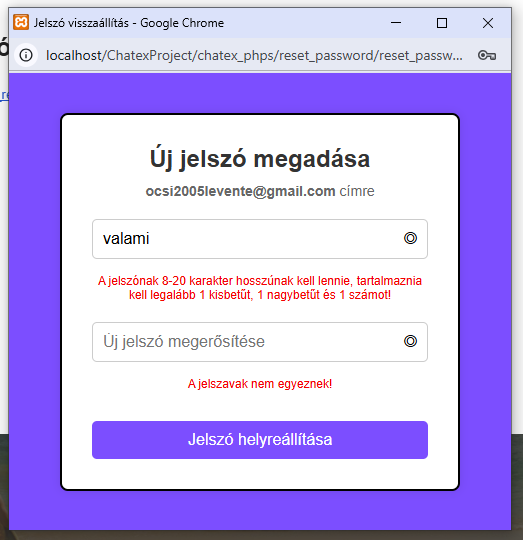
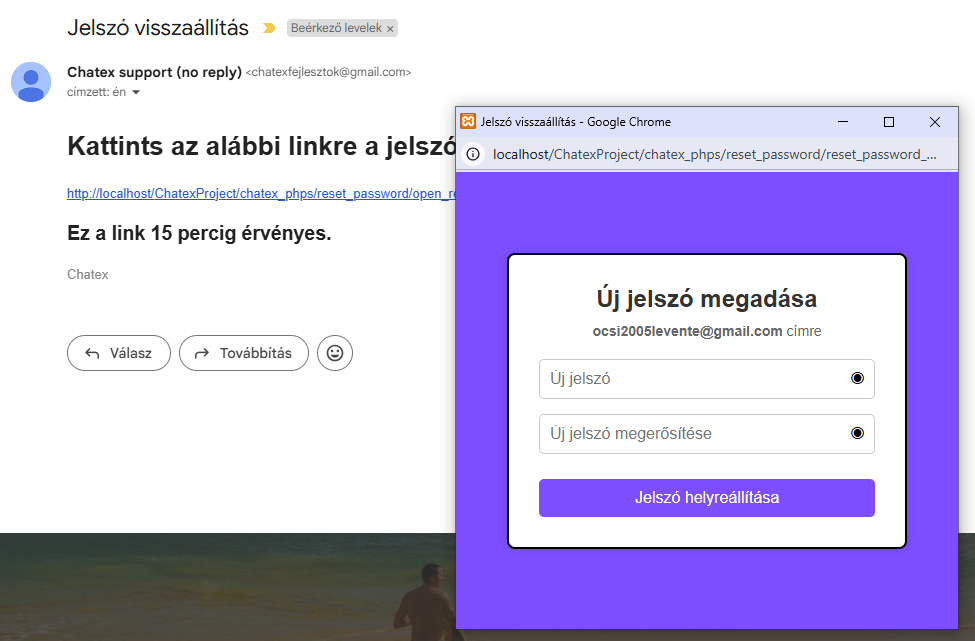
* + Megnyílik egy oldal, ahol a felhasználó beírhatja az új jelszavát.

Ha a token lejárt vagy hibás:

* + Egy hibaüzenetes oldal jelenik meg.

A weboldal tartalmaz:

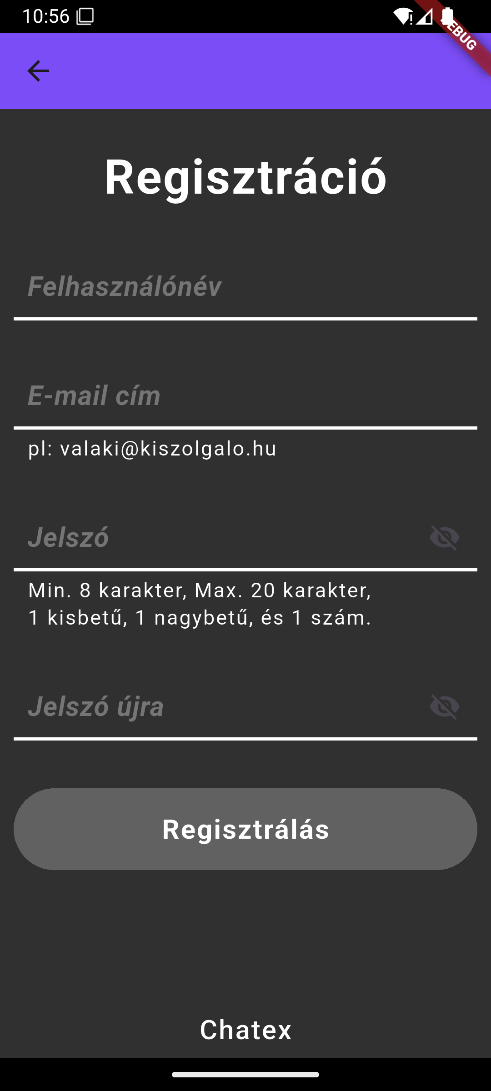
* Két jelszó mezőt (az új jelszó és annak megerősítése).
* Ellenőrzést, hogy a jelszó megfelelő-e:
  + Legalább 8 karakter hosszú legyen.
  + Legyen benne kis- és nagybetű.
  + Tartalmazzon legalább egy számot.
* Hibaüzeneteket, ha valami nem stimmel.



# 5 - Specifikáció

* Főoldal

Az alkalmazás/weboldal megnyitása után megjelenő felület (ha a felhasználó nincs bejelentkezve).

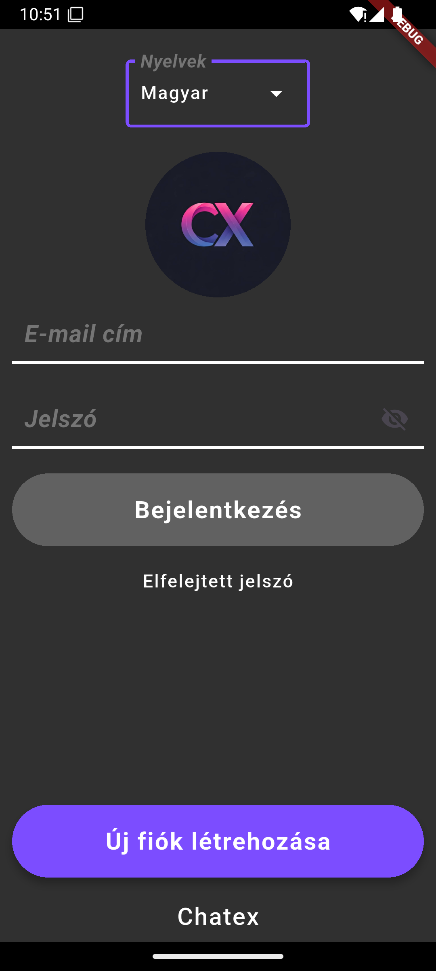
A képen szöveg, képernyőkép, Betűtípus, szoftver látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.A felületen található:

Regisztrációs felület ami tartalmaz:

* Felhasználónév
* Email cím
* Jelszó
* Jelszó megerősítése
* Regisztrálás gomb
* Jelszó megjelenítése
* Megváltozik a mező színe ha nem   
  helyes adatot adnak meg
* Toastmessage-el jelzi hogy sikeres-e a regisztráció

A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.Bejelentkezési felület ami tartalmaz:

* Email cím
* Jelszó
* Bejelentkezés gomb
* Jelszót megjelenítése
* elfelejtett jelszó mező
* Nyelv megváltoztatása
* Toastmessage-el jelzi hogy sikeres-e a bejelentkezés

A képen szöveg, képernyőkép, Betűtípus, szoftver látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.A képen szöveg, képernyőkép, szoftver, Betűtípus látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.Elfelejtett jelszó felület:

1. Megadja a felhasználó az email címét
2. Megnyomja a Jelszó helyreállítása gombot
3. Ha helyes email címet adott meg, akkor az email fiókjában találja az üzenetet az új jelszó kéréséről.

A főképernyőn fér hozzá a felhasználó az alkalmazásunk fő részeihez  
mint például maga a chatekhez, a csoportokhoz ahol egyszerre több személy tud

A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.egy helyen chatelni egymással, beállítások menü ahol minden fontosat beállíthat, Ismerősök kezelése és jelölése.

A képen szöveg, képernyőkép, multimédia, szoftver látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.

A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.

A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.

A képen képernyőkép, szöveg, szoftver, Multimédiás szoftver látható

Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.