BGSzC Pestszentlőrinci Közgazdasági és Informatikai Szakgimnáziuma

1184 Budapest Hengersor 34.

1. Záró dolgozat

Küzdő Kert

Konzulens tanár:

Dobrocsi Róbertné Csilla

Készítette:Pekny Márk,Mózer Richárd,Rapcsák Marcell

Tartalom

[1 Bevezetés 3](#_Toc85723173)

[1.1 Feladat leírás 3](#_Toc85723174)

[1.2 A felhasznált ismeretek 3](#_Toc85723175)

[1.3 A felhasznált szoftverek 3](#_Toc85723176)

[2 Felhasználói dokumentáció 4](#_Toc85723177)

[2.1 A program általános specifikációja 4](#_Toc85723178)

[2.2 Rendszerkövetelmények 4](#_Toc85723179)

[2.2.1 Hardver követelmények 4](#_Toc85723180)

[2.2.2 Szoftver követelmények 4](#_Toc85723181)

[2.3 3. A program telepítése 4](#_Toc85723182)

[2.4 A program használatának a részletes leírása 5](#_Toc85723183)

[3 Fejlesztői dokumentáció 6](#_Toc85723184)

[3.1 Az alkalmazott fejlesztői eszközök 6](#_Toc85723185)

[3.2 Adatmodell leírása 6](#_Toc85723186)

[3.3 Részletes feladatspecifikáció, algoritmusok 6](#_Toc85723187)

[3.4 Tesztelési dokumentáció 7](#_Toc85723188)

[4 Összefoglalás 8](#_Toc85723189)

[4.1 Önértékelés 8](#_Toc85723190)

[4.2 Továbbfejlesztési lehetőségek 8](#_Toc85723191)

[5 Felhasznált irodalom 9](#_Toc85723192)

[6 Ábrajegyzék 10](#_Toc85723193)

# Bevezetés

A "Küzdősportok Közösségi Platformja" egy webalkalmazás, amelyet a küzdősportok iránt érdeklődő közösség számára fejlesztettünk. A célunk egy olyan platform létrehozása volt, ahol a felhasználók edzéseket, eseményeket, ranglistákat követhetnek nyomon, és élő közvetítéseket nézhetnek amiket az edzők közvetítenek. A program React alapú frontenddel, Node.js alapú backenddel, MySQL adatbázissal. A dokumentáció célja, hogy részletesen bemutassa a program telepítését, használatát, fejlesztési folyamatát, valamint a tesztelési és továbbfejlesztési lehetőségeket.

## Feladat leírás

A felhasználói dokumentáció célja, hogy a "Küzdősportok Közösségi Platformja" alkalmazás telepítésével és használatával kapcsolatos minden információt biztosítson a felhasználók számára. A dokumentáció segít a program kezelésének elsajátításában, a rendszerkövetelmények megértésében, a telepítési folyamat követésében, valamint a program funkcióinak részletes megismerésében. A fejlesztői dokumentáció pedig a program logikájának és kódjának megértését, valamint a további fejlesztést támogatja.

**A rendszerrel szemben támasztott főbb funkcionális követelmények:**

1. **Felhasználókezelés:**
   * Regisztráció (látogató/edző szerepkör választással).
   * Biztonságos bejelentkezés (hashelt jelszavak).
   * Profiladatok megtekintése és módosítása.
   * Jelszóváltoztatás.
   * Kijelentkezés.
2. **Információ és Böngészés:**
   * Különböző küzdősportok bemutatása (leírás, szabályok).
   * Sportágakhoz tartozó klubok listázása, részletes adatokkal.
   * Közelgő események (versenyek, táborok) naptáras megjelenítése, részletes információkkal és helyszínnel.
   * Legfrissebb küzdősport hírek megjelenítése külső API-ból.
3. **Közösségi és Interaktív Funkciók:**
   * Klubok értékelésének lehetősége (csillagos és szöveges).
   * Ranglista megjelenítése (legaktívabb edzők és látogatók).
   * Edzésekre való online jelentkezés és lemondás.
4. **Szerepkör-specifikus Funkciók:**
   * **Edzőknek:** Klubok és edzések létrehozása, kezelése (törlés). Élő stream indítása és leállítása.
   * **Látogatóknak:** Jelentkezett edzések nyomon követése (Edzésnapló), statisztikák, motiváció.
5. **Valós idejű Funkciók:**
   * Élő stream állapotának jelzése.
   * Felhasználói értesítések megjelenítése (új jelentkezés, sikeres jelentkezés stb.).

**Technikai követelmények:**

* Frontend: React alapú, komponens-vezérelt architektúra. Reszponzív dizájn.
* Backend: Node.js és REST API.
* Adatbázis: MySQL relációs adatbázis.
* Biztonság: Jelszavak biztonságos tárolása (hash), session/cookie alapú authentikáció.
* Kódminőség: Olvasható, karbantartható kód, verziókezelés (Git/GitHub) használata.

## A felhasznált ismeretek

A "Küzdő Kert" projekt sikeres megvalósításához széles körű informatikai ismeretekre és technológiai jártasságra volt szükségünk. A fejlesztés során az alábbi főbb területeken alkalmaztuk és mélyítettük el tudásunkat:

* **Webfejlesztési alapok:**
  + **HTML5:** A weboldalak tartalmának strukturálása, szemantikus elemek használata.
  + **CSS3:** A felhasználói felület stílusának kialakítása, elrendezések (Flexbox, Grid), animációk, reszponzív dizájn megvalósítása media query-k segítségével.
  + **JavaScript (ES6+):** A dinamikus funkcionalitás implementálása mind a frontend, mind a backend oldalon. Aszinkron programozás (Promises, async/await) az API hívások és időzített feladatok kezelésére. DOM manipuláció (bár React esetében ez közvetettebb).
* **Frontend fejlesztés (React ):**
  + **React:** Komponens alapú fejlesztés, JSX szintaxis, state (állapot) és props (tulajdonságok) kezelése, életciklus-metódusok (funkcionális komponenseknél useEffect hook), eseménykezelés.
  + **React Router:** Egyoldalas alkalmazás (SPA) navigációjának megvalósítása, dinamikus útvonalak (/SportLeiras/:id, /Klub/:id) kezelése useParams hookkal, programozott navigáció useNavigate hookkal.
  + **Állapotkezelés:** useState hook alapvető állapotokhoz, komplexebb állapotok kezelése objektumokkal és tömbökkel.
  + **API kommunikáció:** Axios könyvtár használata HTTP kérések (GET, POST, PUT, DELETE) küldésére a backend API felé, válaszok és hibák kezelése. withCredentials: true beállítás a sütik küldéséhez.
  + **UI könyvtárak és stílus:** Bootstrap integrálása a gyorsabb UI fejlesztéshez és reszponzivitáshoz. Egyedi CSS modulok/fájlok használata a specifikus stílusokhoz. React Icons az ikonok beillesztéséhez. SweetAlert2 a felhasználói visszajelzésekhez. React Calendar a naptár funkcióhoz.
* **Backend fejlesztés (Node.js ökoszisztéma):**
  + **Node.js:** Szerveroldali JavaScript futtatása.
  + **Express.js:** RESTful API végpontok definiálása, útválasztás (routing), middleware-ek használata (pl. cors, body-parser - implicit az Express újabb verzióiban, authentikációs middleware).
  + **Adatbázis-kezelés:** MySQL adatbázis-kapcsolat felépítése, SQL lekérdezések (SELECT, INSERT, UPDATE, DELETE) futtatása Node.js alatt (valószínűleg mysql2 csomaggal). Paraméterezett lekérdezések használata az SQL injection megelőzésére.
  + **Authentikáció és Autorizáció:** Session-alapú bejelentkezés kezelése (valószínűleg express-session és cookie-parser segítségével), jelszavak biztonságos tárolása bcrypt segítségével (hashelés és ellenőrzés). Szerepkör alapú hozzáférés-vezérlés implementálása bizonyos API végpontokhoz.
  + **Valós idejű kommunikáció:** Socket.IO használata a szerver és kliens közötti kétirányú, valós idejű kommunikációra (pl. LiveStream állapotának frissítése).
* **Adatbázis-tervezés és -kezelés:**
  + **MySQL:** Relációs adatbázis-tervezés alapelvei (normalizálás alapszinten), táblák, mezők, adattípusok definiálása. Elsődleges és idegen kulcsok használata a táblák közötti kapcsolatok és az adatbázis integritásának biztosítására. Indexek létrehozása a lekérdezések gyorsítására.
  + **SQL:** Adatdefiníciós (DDL) és adatmanipulációs (DML) parancsok írása. JOIN műveletek használata több táblából származó adatok összekapcsolására. Adatbázis triggerek írása automatizált feladatokhoz.
  + **MySQL Workbench:** Vizuális eszköz az adatbázis tervezéséhez, kezeléséhez, lekérdezések teszteléséhez.
* **Verziókezelés és fejlesztési eszközök:**
  + **Git & GitHub:** Kódverziók követése, ágkezelés (branching), változások egyesítése (merging), konfliktuskezelés. Közös munka támogatása GitHub platformon keresztül.
  + **Fejlesztői környezet:** Visual Studio Code hatékony használata (kiterjesztések, debuggolás alapszinten).
  + **Csomagkezelés:** npm használata a projekt függőségeinek telepítésére és frissítésére (package.json, package-lock.json).
  + **API tesztelés:** Postman használata a backend API végpontjainak manuális tesztelésére.

Ezen ismeretek tették lehetővé a "Küzdő Kert" platform megtervezését, fejlesztését és tesztelését. A projekt során mindannyian jelentős gyakorlati tapasztalatot szereztünk ezeken a területeken.

## A felhasznált szoftverek

A projekt fejlesztése, tesztelése és futtatása során az alábbi szoftvereszközök és technológiák kerültek felhasználásra. Ezek biztosították a szükséges keretrendszereket, könyvtárakat és segédeszközöket a "Küzdő Kert" platform létrehozásához.

Integrált Fejlesztői Környezet (IDE) / Kódszerkesztő:

Visual Studio Code (VS Code): Ingyenes, platformfüggetlen, nagymértékben testreszabható kódszerkesztő, amely kiválóan támogatja a JavaScript, React, Node.js fejlesztést számos beépített funkcióval és kiterjesztéssel (pl. IntelliSense, hibakeresés, Git integráció).

Futtatókörnyezet:

Node.js (v18.x vagy újabb): JavaScript futtatókörnyezet, amely lehetővé teszi a JavaScript futtatását a böngészőn kívül, a szerveroldalon. Tartalmazza az npm csomagkezelőt.

Csomagkezelő: npm (Node Package Manager):

A Node.js alapértelmezett csomagkezelője, amely a projekt külső függőségeinek (könyvtárak, keretrendszerek) telepítését és kezelését végzi (`package.json`).

Adatbázis-kezelő Rendszer:

MySQL (v8.x vagy újabb): Nyílt forráskódú, relációs adatbázis-kezelő rendszer, amelyet az alkalmazás adatainak tárolására használtunk.

Adatbázis Menedzsment Eszköz: MySQL Workbench: Hivatalos, grafikus eszköz a MySQL adatbázisok tervezéséhez, fejlesztéséhez, adminisztrációjához és SQL lekérdezések futtatásához.

Verziókezelő Rendszer: Git: Elosztott verziókezelő rendszer a forráskód változásainak nyomon követésére, a csapattagok közötti együttműködés megkönnyítésére.

GitHub: Webalapú hoszting szolgáltatás Git repozitóriumok számára, amely a projekt központi kód- és projektmenedzsment platformjaként funkcionált.

GitHub Desktop: Grafikus kliens a Git és GitHub használatához.

API Tesztelő Eszköz: Postman: Platform API-k fejlesztéséhez és teszteléséhez, lehetővé teszi HTTP kérések egyszerű küldését és a válaszok elemzését.

Frontend Keretrendszer/Könyvtár: React (v18.x): Deklaratív, komponensalapú JavaScript könyvtár felhasználói felületek építésére.

React Router (v6): Kliensoldali útválasztási megoldás React alkalmazásokhoz.

Bootstrap (v5.x): Népszerű, előre elkészített komponenseket és reszponzív rácsrendszert biztosító CSS/JS keretrendszer.

Axios: HTTP kliens könyvtár a backend API-val való kommunikációhoz.

SweetAlert2: Könyvtár esztétikus és interaktív felugró üzenetek (alert, confirm, prompt) megjelenítésére.

React Calendar: Komponens naptárak egyszerű integrálásához.

React Icons: Különböző ikoncsomagok (pl. Font Awesome) egyszerű használata React komponensként.

Framer Motion: Animációs könyvtár React komponensekhez.

Dokumentumfejléc (head) elemeinek dinamikus kezelése React komponensekből.

Backend Keretrendszer/Könyvtárak: Webalkalmazás-keretrendszer Node.js-hez, megkönnyíti a REST API-k és webalkalmazások fejlesztését.

mysql2 (vagy hasonló Node.js MySQL driver): Könyvtár a Node.js alkalmazás és a MySQL adatbázis közötti kapcsolat és kommunikáció kezelésére.

bcrypt: Könyvtár jelszavak biztonságos hashelésére és ellenőrzésére.

cors: Middleware a Cross-Origin Resource Sharing (CORS) policy-k kezelésére, lehetővé téve a frontend számára a backend API elérését más domainről/portról.

express-session / cookie-parser (feltételezett): Middleware-ek a felhasználói munkamenetek (session) kezelésére sütik (cookie) segítségével.

Build Eszköz (Frontend): Vite: Modern, gyors frontend build eszköz, amely fejlesztés közben azonnali Hot Module Replacement (HMR) funkciót, produkcióban pedig optimalizált csomagokat biztosít.

Webböngészők: Google Chrome, , Microsoft Edge: A fejlesztéshez, teszteléshez és a végfelhasználói használathoz.

Ezek a szoftverek és könyvtárak alkották azt a technológiai alapot, amelyre a "Küzdő Kert" platform épült.

# Felhasználói dokumentáció

Ez a fejezet a "Küzdősportok Közösségi Platformja" ("Küzdő Kert") webalkalmazás használatához nyújt segítséget a végfelhasználók számára.

## A program általános specifikációja

A "Küzdősportok Közösségi Platformja" egy webalkalmazás, amely a következő funkciókat kínálja:

**Edzésnapló**: A felhasználók nyomon követhetik, mely edzésekre jelentkeztek, és lemondhatják azokat.Nyomon követheted hogy hány edzésen vettél részt és motivációs szöveget is kapsz.

**Ranglista**: Az edzők rangsorának megtekintése hozzá adott edzések alapján.

**Livestream**: Élő közvetítések megtekintése. A program modern, reszponzív dizájnnal rendelkezik, és alkalmas közösségi interakciók támogatására.

**Profil**: A profil oldalon módosíthatod a saját adataidat, és beállíthatod, hogy milyen típusú edzések érdekelnek téged. Így személyre szabott edzéseket láthatsz.

**Események**: Itt láthatod milyen közelgő versenyek lesznek részletes leírást róla(Milyen fajta küzdő sportban lesz kik vehetnek részt és hogy hol lesz a verseny amit google maps segítségével is biztosítunk).

**Edzői oldal**:Itt kezelheted az edzéseid és klubbod adhatsz hozzá új edzést illetve törölhetsz is,livestreamet innen tudod kezelni a youtube linkeddel.

Az alkalmazás két fő felhasználói szerepkört különböztet meg:

1**. Látogató** (Visitor): Olyan felhasználó, aki információt keres, edzésekre jelentkezik, eseményeket követ, értékel.

2. **Edző** (Coach): Olyan felhasználó, aki a látogatói funkciók mellett saját tartalmat (klub, edzés, stream) is létrehozhat és kezelhet.

## Rendszerkövetelmények

### Hardver követelmények

Minimális konfiguráció:

Processzor: 1 GHz vagy gyorsabb.

RAM: 2 GB.

Tárhely: 500 MB szabad hely.

Internetkapcsolat: Stabil kapcsolat (legalább 1 Mbps).

Ajánlott konfiguráció:

Processzor: 2 GHz vagy gyorsabb.

RAM: 4 GB vagy több.

Tárhely: 1 GB szabad hely.

Internetkapcsolat: 5 Mbps vagy gyorsabb.

### Szoftver követelmények

Operációs rendszer:

Windows 10/11, macOS 10.15 vagy újabb, Linux (Ubuntu 20.04 vagy újabb).

Böngésző:

Google Chrome, Mozilla Firefox, Microsoft Edge (legfrissebb verzió).

Szükséges szoftverek:

Node.js (v18.x vagy újabb).

MySQL (v8.x vagy újabb).

Git (verziókezeléshez).

Adatbázis:

MySQL adatbázis (kuzdosportok nevű adatbázis).

## 3. A program telepítése

Ez a szakasz a fejlesztőknek vagy azoknak szól, akik a programot saját számítógépükön szeretnék futtatni és esetleg módosítani. **A normál felhasználóknak erre a lépésre nincs szükségük**, ők az alkalmazást a megadott webcímen (URL) keresztül érik el.

**Előfeltételek:**

1. **Node.js és npm:** Telepítsd a Node.js 18.x vagy újabb verzióját ([https://nodejs.org/](https://www.google.com/url?sa=E&q=https%3A%2F%2Fnodejs.org%2F)). Ellenőrizd a telepítést a parancssorban: node -v és npm -v.
2. **MySQL:** Telepítsd a MySQL 8.x vagy újabb verzióját ([https://dev.mysql.com/downloads/mysql/](https://www.google.com/url?sa=E&q=https%3A%2F%2Fdev.mysql.com%2Fdownloads%2Fmysql%2F)). A telepítés során jegyezd meg vagy állítsd be a root felhasználó jelszavát.
3. **Git:** Telepítsd a Git verziókezelőt ([https://git-scm.com/](https://www.google.com/url?sa=E&q=https%3A%2F%2Fgit-scm.com%2F)). Ellenőrizd: git --version.
4. **(Opcionális) MySQL Workbench:** Grafikus MySQL kliens az adatbázis kezeléséhez ([https://dev.mysql.com/downloads/workbench/](https://www.google.com/url?sa=E&q=https%3A%2F%2Fdev.mysql.com%2Fdownloads%2Fworkbench%2F)).
5. **(Opcionális) GitHub Desktop:** Grafikus Git kliens ([https://desktop.github.com/](https://www.google.com/url?sa=E&q=https%3A%2F%2Fdesktop.github.com%2F)).

**Telepítési lépések:**

1. **Projekt letöltése (Klónozás):**
   * Nyiss egy parancssort vagy terminált.
   * Navigálj abba a könyvtárba, ahová a projektet telepíteni szeretnéd.
   * Klónozd a projekt GitHub repozitóriumát (a megadott link alapján):

git clone https://github.com/zaromunka\_kuzdosportok/React/KuzdoKert.git

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487" \t "_blank).Bash

* + Lépj be a letöltött KuzdoKert mappába:

cd KuzdoKert

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487" \t "_blank).Bash

1. **Adatbázis beállítása:**
   * Indítsd el a MySQL szervert.
   * Jelentkezz be a MySQL-be (pl. MySQL Workbench vagy mysql -u root -p).
   * Hozd létre az adatbázist:

CREATE SCHEMA IF NOT EXISTS `kuzdosportok` DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4\_hungarian\_ci;

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487" \t "_blank).SQL

* + Importáld a projektben található josql8.sql (vagy az aktuális nevű SQL fájl) tartalmát a kuzdosportok adatbázisba.
    - **MySQL Workbench:** Server -> Data Import -> Import from Self-Contained File, válaszd ki az SQL fájlt, cél sémának a kuzdosportok-ot, majd Start Import.
    - **Parancssor:** (A KuzdoKert mappából)

mysql -u root -p kuzdosportok < josql8.sql

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487" \t "_blank).Bash

(Add meg a jelszót, ha szükséges.)

* + **Fontos ellenőrzés:** Győződj meg róla, hogy a notifications táblában létezik a read TINYINT(1) NOT NULL DEFAULT 0 oszlop. Ha nem, add hozzá:
  + USE kuzdosportok;

ALTER TABLE `notifications` ADD COLUMN `read` TINYINT(1) NOT NULL DEFAULT 0 AFTER `message`;

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487" \t "_blank).SQL

1. **Backend függőségek telepítése és indítása:**
   * Navigálj a Backend mappába:

cd Backend

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487" \t "_blank).Bash

* + Telepítsd a szükséges Node.js csomagokat:

npm install

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487" \t "_blank).Bash

* + Indítsd el a backend szervert (fejlesztői módban, ami általában automatikusan újraindul változáskor):

npm run dev

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487" \t "_blank).Bash

*(Figyeld a konzolt a szerver indulásáról és a portról, pl. 5000)*

1. **Frontend függőségek telepítése és indítása:**
   * Nyiss egy *új* parancssort/terminált (hagyd a backendet futni).
   * Navigálj a Frontend mappába:

cd ../Frontend

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487" \t "_blank).Bash

* + Telepítsd a szükséges Node.js csomagokat:

npm install

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487" \t "_blank).Bash

* + Indítsd el a Vite fejlesztői szervert:

npm run dev

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487" \t "_blank).Bash

*(Figyeld a konzolt a frontend elérhetőségéről, pl.*[*http://localhost:5173*](https://www.google.com/url?sa=E&q=http%3A%2F%2Flocalhost%3A5173)*)*

1. **Alkalmazás elérése:**
   * Nyisd meg egy webböngészőt.
   * Írd be a címsorba a frontend fejlesztői szerver címét (pl. http://localhost:5173).
   * Az alkalmazásnak be kell töltődnie. Regisztrálhatsz, vagy bejelentkezhetsz a josql8.sql-ben található felhasználókkal (figyelve a jelszavakra - ha nem ismertek, regisztrálj újat).

## A program használatának a részletes leírása

Regisztrációnál kell megadni, hogy edzőként vagy látogatóként vagy jelen a weboldalon. Ezt követi a bejelentkezés. Amennyiben edzőként regisztrál az illető edzői oldalt talál az illető, ahol hozzátudja adni a saját klubját és az edzéseit a weboldalhoz. A látogató az edzői oldal helyett edzésnaplót talál, ahol azokat az edzéseket látja amikre már jelentkezett, valamint a kitűzött célját és az elért eredményét és még két motivációs szöveget. A sportok fülnél kiválasztott sportágnál meglévő klubbokat értékelni tudja csillagozással és szöveges értékeléssel egyaránt.

**2.4.1. Regisztráció és Bejelentkezés**

* **Regisztráció:**
  1. Az oldal tetején található navigációs sávban kattints a "Regisztráció" linkre.
  2. A megjelenő űrlapon add meg a kért adatokat. A csillaggal (\*) jelölt mezők kitöltése kötelező (Vezetéknév, Keresztnév, Email, Szül.idő, Lakhely, Felhasználónév, Jelszó, Jelszó ismét). A harmadik név és a telefonszám opcionális.
  3. Válaszd ki a kívánt **Szerepkört** (Látogató vagy Edző).
  4. Olvasd el és fogadd el az **Általános Szerződési Feltételeket** a jelölőnégyzet bepipálásával.
  5. Kattints a "Regisztráció" gombra. Sikeres regisztráció esetén üzenetet kapsz, és a bejelentkezési oldalra kerülsz. Hiba esetén (pl. foglalt név, rossz jelszó) az űrlapon hibaüzenet jelenik meg.
* **Bejelentkezés:**
  1. A navigációs sávban kattints a "Bejelentkezés" linkre.
  2. Add meg a regisztrációkor választott **Felhasználónevedet** és **Jelszavadat**.
  3. Kattints a "Bejelentkezés" gombra.
  4. Sikeres azonosítás után betöltődik az Üdvözlő képernyő, majd a Főoldal. A navigációs sáv frissül a bejelentkezett felhasználó opcióival. Hibás adatok esetén hibaüzenetet kapsz.

**2.4.2. Navigáció és Főoldal**

* **Navigációs Sáv:** A felső menü segítségével érheted el a fő oldalakat (Főoldal, Sportok, Események, Élő Stream, Ranglista, Edzésnapló/Edzői oldal, Profil stb.). A menü tartalma függ a bejelentkezési státusztól és a szerepkörtől.
* **Főoldal (/)**: Az alkalmazás kezdőoldala. Itt találsz bemutatkozó szövegeket, legfrissebb küzdősport híreket (Box, Kickbox, MMA), információkat a platformról, és felhasználói véleményeket.

**2.4.3. Sportok és Klubok Böngészése, Értékelés**

1. Kattints a "Sportok" menüpontra.
2. Válassz egy sportágat a megjelenő kártyák közül.
3. A sportág részletes oldalán (SportLeiras):
   * Olvashatsz a sportágról és a szabályairól.
   * Láthatod a kapcsolódó közelgő eseményeket.
   * Böngészhetsz a sportághoz tartozó klubok között.
   * Megtekintheted mások értékeléseit a klubokról.
   * **Értékelés (csak Látogatók):** Adj csillagos (1-5) és szöveges értékelést a klubhoz, majd kattints a "Értékelés beküldése" gombra.
   * Kattints a "További információk..." gombra egy klubnál, hogy a klub saját oldalára (Klub) juss.

**2.4.4. Edzésre Jelentkezés és Lemondás**

1. Navigálj egy klub részletes oldalára (Klub).
2. Az "Edzések" listában keresd meg a kívánt edzést.
3. **Jelentkezés:** Kattints a "Jelentkezés" gombra (ha be vagy jelentkezve látogatóként és nem saját edzésed). Sikeres jelentkezésről üzenet értesít, a gomb "Visszavonás"-ra vált.
4. **Lemondás:** Kattints a "Visszavonás" gombra, ha már jelentkeztél. Sikeres lemondás után a gomb újra "Jelentkezés" lesz.

**2.4.5. Edzésnapló használata**

1. Kattints az "Edzésnapló" menüpontra (bejelentkezés után).
2. **Jelentkezett edzéseid:** Látod a listát az aktív jelentkezéseidről (klub, sport, hely, idő). A "Lemondás" gombbal törölheted a jelentkezést.
3. **Fejlődés:** Láthatod a részt vett edzések számát, a sportágak szerinti eloszlást, és a havi célodhoz viszonyított haladást egy sávon.
4. **Motiváció:** Rövid, bátorító üzeneteket olvashatsz.

**2.4.6. Edzői Oldal funkciói (csak Edzőknek)**

1. Kattints az "Edzői oldal" menüpontra (ha edzőként vagy bejelentkezve).
2. **Klub hozzáadása:** Töltsd ki az űrlapot (Sport, Klubb név, Város, Szabályok, Leírás), majd kattints a "Klubb hozzáadása" gombra.
3. **Edzés hozzáadása:** Válaszd ki a klubodat, add meg a pontos címet, napot, időpontot, majd kattints az "Edzés hozzáadása" gombra.
4. **Kezelés:** Az oldal alján listázva látod a klubjaidat és edzéseidet. A "Törlés" gombbal eltávolíthatod őket (klub törlése az edzéseit is törli).
5. **Élő Stream:** Add meg a YouTube linket az "Stream URL" mezőbe. Indítsd ("Stream indítása") vagy állítsd le ("Stream leállítása") a közvetítést az "Élő Stream" oldalon.

**2.4.7. Események megtekintése**

1. Kattints az "Események" menüpontra.
2. A naptárban kattints egy kiemelt napra.
3. A naptár alatt megjelennek az aznapi események részletei (leírás, helyszín linkkel, időpont, státusz, szervező).

**2.4.8. Élő Stream nézése**

1. Kattints az "Élő Stream" menüpontra.
2. Ha egy edző éppen közvetít, a videólejátszó megjelenik. Ha nem, egy üzenet jelzi ezt.

**2.4.9. Ranglista használata**

1. Kattints a "Ranglista" menüpontra.
2. Válassz szűrőt ("Összes" / "Utolsó 30 nap").
3. Látod a legaktívabb edzők (legtöbb edzés) és látogatók (legtöbb részvétel) rangsorát. Az első három helyezett ki van emelve.
4. Használd a lapozó gombokat ("Előző", "Következő"), ha a lista több oldalas.

**2.4.10. Profil kezelése**

1. Kattints a profil ikonra (jobb felső sarok), majd a "Profilom" linkre.
2. **Adatok:** Látod az adataidat. Kattints a "Szerkesztés" gombra a módosításhoz, majd a "Mentés" gombra a változtatások elmentéséhez.
3. **Jelszó:** A "Jelszó változtatása" részben add meg a régi és az új jelszót (kétszer), majd kattints a "Jelszó változtatása" gombra.

**2.4.11. Értesítések kezelése**

1. Figyeld a **csengő (🔔) ikont** a navigációs sávban. A szám jelzi az olvasatlan értesítéseket.
2. Vidd az egeret az ikon fölé (vagy kattints rá).
3. A lenyíló listában látod az értesítéseket. Megtekintéskor automatikusan olvasottá válnak.

**2.4.12. Kijelentkezés**

1. Kattints a profil ikonra, majd a "Kijelentkezés" linkre.
2. A rendszer kijelentkeztet és a Bejelentkezés oldalra irányít.

# Fejlesztői dokumentáció

A fejlesztői dokumentáció célja, hogy a segítse program logikájának, illetve a program kódjának a megértését, illetve a program továbbfejlesztését. Szakemberek számára készül, elvárás tehát a pontosság és a szakmai jellegű stílus.

## Az alkalmazott fejlesztői eszközök

A projekt fejlesztése, tesztelése és futtatása során az alábbi szoftvereszközök és technológiák kerültek felhasználásra. Ezek biztosították a szükséges keretrendszereket, könyvtárakat és segédeszközöket a "Küzdő Kert" platform létrehozásához.

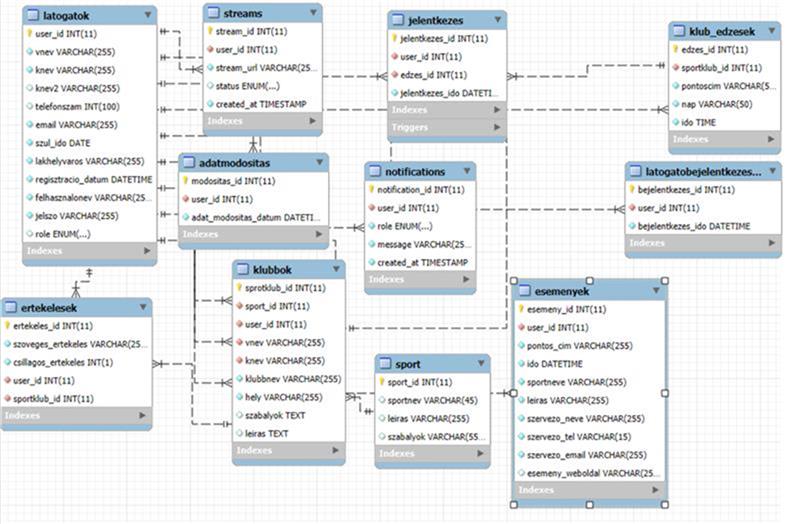
* **Integrált Fejlesztői Környezet (IDE) / Kódszerkesztő:**
  + **Visual Studio Code (VS Code):** Ingyenes, platformfüggetlen, nagymértékben testreszabható kódszerkesztő.
* **Futtatókörnyezet:**
  + **Node.js (v18.x vagy újabb):** JavaScript futtatókörnyezet a szerveroldalhoz.
* **Csomagkezelő:**
  + **npm (Node Package Manager):** Függőségek telepítésére és kezelésére (package.json).
* **Adatbázis-kezelő Rendszer:**
  + **MySQL (v8.x vagy újabb):** Relációs adatbázis-kezelő.
* **Adatbázis Menedzsment Eszköz:**
  + **MySQL Workbench:** Grafikus eszköz adatbázisok tervezéséhez, kezeléséhez.
* **Verziókezelő Rendszer:**
  + **Git:** Forráskód változásainak nyomon követésére.
  + **GitHub:** Webalapú Git repozitórium hoszting.
  + **GitHub Desktop:** Grafikus Git kliens.
* **API Tesztelő Eszköz:**
  + **Postman:** Backend API végpontok manuális tesztelésére.
* **Frontend Keretrendszer/Könyvtár:**
  + **React (v18.x):** Felhasználói felületek építésére.
  + **React Router (v6):** Kliensoldali útválasztás.
  + **Bootstrap (v5.x):** CSS/JS keretrendszer.
  + **Axios:** HTTP kliens.
  + **SweetAlert2:** Felugró üzenetek.
  + **React Calendar:** Naptár komponens.
  + **React Icons:** Ikonok használata.
  + **Socket.IO Client:** Valós idejű kommunikáció (kliens).
  + **Framer Motion:** Animációk.
  + **react-helmet:** <head> elemek kezelése.
* **Backend Keretrendszer/Könyvtárak:**
  + **Express.js:** Webalkalmazás-keretrendszer Node.js-hez.
  + **mysql2 (vagy hasonló):** Node.js MySQL driver.
  + **bcrypt:** Jelszó hashelés.
  + **cors:** CORS policy kezelése.
  + **express-session / cookie-parser (feltételezett):** Session kezelés.
  + **Socket.IO Server:** Valós idejű kommunikáció (szerver).
* **Build Eszköz (Frontend):**
  + **Vite:** Gyors frontend build eszköz és fejlesztői szerver.
* **Webböngészők:**
  + Google Chrome, Microsoft Edge: Fejlesztéshez, teszteléshez.

**Felhasznált modulok és jogszerűség:** A projekt számos nyílt forráskódú modult használ (React, Express, Axios stb.), amelyek jellemzően MIT vagy hasonló megengedő licenccel rendelkeznek, lehetővé téve széleskörű felhasználásukat. A külső API-k (News API, Google Translate API - bár utóbbi implementációja kérdéses a kulcs hiánya miatt) saját felhasználási feltételekkel bírnak, amelyeket a szolgáltatók oldalán kell ellenőrizni (pl. ingyenes kvóták, kereskedelmi felhasználás korlátai). A fejlesztők felelőssége ezen licencek betartása.

## Adatmodell leírása

Az alkalmazás adatait a kuzdosportok nevű MySQL relációs adatbázis tárolja. Az alábbiakban bemutatjuk a séma felépítését és a táblák közötti kapcsolatokat.

**1.Ábra**



**Táblák leírása:**

1. **sport:** Sportágak alapadatai (ID, név, leírás, szabályok).
2. **latogatok:** Regisztrált felhasználók (ID, név, elérhetőségek, belépési adatok, szerepkör).
3. **klubbok:** Edzők által létrehozott klubok (ID, sportág ID, edző ID, klub adatai).
4. **klub\_edzesek:** Klubokhoz tartozó edzések (ID, klub ID, pontos cím, nap, idő).
5. **jelentkezes:** Felhasználók jelentkezései edzésekre (ID, felhasználó ID, edzés ID, jelentkezés ideje).
6. **ertekelesek:** Klubok felhasználói értékelései (ID, szöveg, csillagok, felhasználó ID, klub ID, dátum).
7. **esemenyek:** Küzdősport események (ID, létrehozó ID, helyszín, idő, sportnév, leírás, szervező adatai).
8. **streams:** Élő közvetítések adatai (ID, edző ID, URL, státusz, frissítés ideje).
9. **notifications:** Felhasználói értesítések (ID, címzett ID, üzenet, olvasott állapot, létrehozás ideje). **Fontos:** A read oszlopot manuálisan hozzá kell adni a sémához!
10. **adatmodositas:** Profilmódosítások naplózása (ID, felhasználó ID, dátum).
11. **latogatobejelentkezesek:** Bejelentkezések naplózása (ID, felhasználó ID, dátum).

**Főbb kapcsolatok és integritási szabályok:**

* **Kapcsolatok:** Idegen kulcsok kötik össze a táblákat (pl. klubbok.sport\_id -> sport.sport\_id, klub\_edzesek.sportklub\_id -> klubbok.sprotklub\_id).
* **Kaszkádolt műveletek:** Bizonyos törlések automatikusan törlik a kapcsolódó rekordokat (pl. klub törlése törli az edzéseit és értékeléseit).
* **Egyediség:** Bizonyos mezőknek egyedinek kell lenniük (pl. latogatok.email, latogatok.felhasznalonev). Kombinációk is lehetnek egyediek (pl. egy felhasználó egy edzésre csak egyszer jelentkezhet).
* **Kötelező mezők:** NOT NULL megszorítások biztosítják a szükséges adatok meglétét.
* **Trigger:** Az after\_jelentkezes\_insert trigger automatikusan értesítéseket hoz létre.

Ez a struktúra biztosítja az adatok konzisztens és logikus tárolását.

## Részletes feladatspecifikáció, algoritmusok

**2. ábra: Rendszerarchitektúra vázlata (Frontend-Backend-Adatbázis) (Tervezett hely)**

**Rendszerarchitektúra:**  
A "Küzdő Kert" egy **kliens-szerver architektúrájú** webalkalmazás.

* **Frontend (React):** Felelős a felhasználói felületért, interakciókért, API hívásokért. A felhasználó böngészőjében fut.
* **Backend (Node.js):** REST API-t biztosít, kezeli az üzleti logikát, kommunikál az adatbázissal, kezeli az authentikációt és a valós idejű kapcsolatokat. Szerveren fut.
* **Adatbázis (MySQL):** Perzisztensen tárolja az alkalmazás adatait.  
  A kommunikáció a frontend és backend között HTTP kérésekkel (REST API) és WebSocketekkel (Socket.IO) történik.

**Frontend komponensstruktúra és adatfolyam:**  
A frontend React komponensekből épül fel. A App.js a gyökér, tartalmazza a Navbar-t és a React Router által kezelt oldal komponenseket (Home, Login, Profil stb.). Az adatokat az oldalkomponensek általában useEffect hook-kal kérik le a backendről Axios segítségével, és useState-tel tárolják lokálisan. Az állapotok frissítése felhasználói interakciók (pl. gombnyomás) vagy API válaszok hatására történik. Globálisabb adatok (pl. bejelentkezési státusz) a Navbar-ban vannak kezelve.

**Backend API végpontok (főbb csoportok):**  
A backend logikailag csoportosított REST végpontokat kínál a frontend számára:

* Authentikáció (/login, /register, /logout)
* Felhasználókezelés (/updateUser, /changePassword)
* Sportágak, Klubok, Edzések (CRUD műveletek, listázások)
* Jelentkezések (/apply-workout, /edzesnaplo, /jelentkezes)
* Értékelések (/ertekelesek)
* Események (/api/esemenyek)
* Ranglista (/api/ranglista)
* Stream (/api/streams)
* Értesítések (/api/notifications)

**Kulcsfontosságú funkciók logikája (Algoritmusok vázlata):**

* **Authentikáció:** Regisztrációnál jelszó hashelés (bcrypt.hash), bejelentkezésnél hashek összehasonlítása (bcrypt.compare), sikeres bejelentkezés után session létrehozása és süti küldése. Védett végpontok előtt middleware ellenőrzi a session érvényességét és a felhasználó szerepkörét.
* **Értesítési rendszer:** Új jelentkezéskor adatbázis trigger (after\_jelentkezes\_insert) INSERT-el a notifications táblába. Frontend periódikusan lekérdezi (GET /api/notifications/...). Lista megnyitásakor frontend PUT kérést küld (/api/notifications/mark-read/...), backend UPDATE-tel olvasottá teszi az értesítéseket.
* **Ranglista számítás:** Backend (GET /api/ranglista) SQL GROUP BY és COUNT() segítségével számolja az edzők edzéseit (vagy az edzéseikre érkezett jelentkezéseket) és a látogatók jelentkezéseit, ORDER BY rendezi, LIMIT és OFFSET kezeli a lapozást. Az időszűrés (WHERE datum >= ...) a filter paraméter alapján történik.
* **Edzői oldal (Klub/Edzés hozzáadása/törlése):** Backend végpontok előtt middleware ellenőrzi a 'coach' szerepkört. Törlésnél további ellenőrzés, hogy az edző csak a saját tartalmát törölhesse (a törlendő elem tulajdonosának user\_id-ját összeveti a sessionben lévő user\_id-val).
* **Élő Stream kezelés:** Backend (POST /api/streams/start, POST /api/streams/stop) frissíti a streams tábla status és stream\_url mezőit. Sikeres állapotváltás után a backend Socket.IO segítségével emittel üzenetet küld ('stream-update') az összes kapcsolódott kliensnek. A frontend (LiveStream.js) figyeli ezt az üzenetet, és frissíti az iframe tartalmát és láthatóságát.

## Tesztelési dokumentáció

1. Legalább 3 különböző teszteset részletes bemutatása.

* különböző felhasználó tevékenységek esetén hogyan reagált a program
* milyen üzeneteket kaptunk
* mi a teendő az egyes üzenetek esetében

1. Normál teszteset, extrém teszteset (bolondbiztosság tesztelése)
2. A tesztelés során kiderült hibák felsorolása A tesztelési dokumentációból derüljön ki, hogy ismered a különböző tesztelési módszereket (pl. fekete doboz, fehér doboz módszer)

Ajánlott terjedelem: a feladat jellegétől függően 2-5 oldal.

# Összefoglalás

A "Küzdősportok Közösségi Platformja" projekt sikeresen megvalósult, és a kitűzött célokat nagyrészt elértük. A platform lehetővé teszi a felhasználók számára, hogy edzéseket, eseményeket és ranglistákat kövessenek nyomon, valamint élő közvetítéseket nézzenek. A fejlesztés során számos kihívással szembesültünk, például a hírek képeinek betöltésével és a reszponzív dizájn biztosításával, de ezeket a problémákat megoldottuk. A projekt során szerzett tapasztalataink jelentősen hozzájárultak szakmai fejlődésünkhöz.

## Önértékelés

**Mózer Richárd**:

**Élményeim a projekt során**

A Küzdősportok Világa projekt elkészítése során rengeteg új élménnyel gazdagodtam, és úgy érzem, hogy mind szakmailag, mind emberileg sokat fejlődtem. Ez a záródolgozat nem csak egy iskolai feladat volt számomra, hanem egy olyan kihívás, amelyben kipróbálhattam magam, és megtanulhattam, hogyan kell egy valós problémára megoldást találni egy csapat részeként. Az alábbiakban szeretném összefoglalni az élményeimet, a fejlődésemet, valamint azokat a nehézségeket, amelyekkel szembe kellett néznem.

Az egyik legemlékezetesebb pillanat számomra az volt, amikor először láttam, hogy a hírek szekció működik, és a News API-ból érkező cikkek szépen megjelennek az oldalon. Nagyon büszke voltam magamra, mert ez volt az első alkalom, hogy egy külső API-val dolgoztam, és sikerült integrálnom a projektbe. Emellett nagyon élveztem a Ranglista oldal dizájnjának kialakítását, különösen a trófea animációk hozzáadását, mert ez még felhasználó barátabbá tette az oldalunkat.

A Probléma megoldó képességem is nagyon sokat fejlődött de ezt nem csak magamról mondhatom el szerintem hiszen mindnyájan megtanultunk tanulni és először magunktól megoldani a problémákat nem egyből a segítség kérés volt az első lépés erre is büszke vagyok nagyon.

**Ami nem ment jól**

Az időmenedzsmentem nem volt tökéletes. A projekt elején alábecsültem, hogy mennyi időt fog igénybe venni a különböző funkciók implementálása, például a Ranglista oldal szűrési logikája. Emiatt az utolsó hetekben nagyon kapkodnom kellett, hogy minden elkészüljön időben, és ez néha a kód minőségének rovására ment. Ha újra kezdhetném, jobban megtervezném a munkafolyamatot, és több időt hagynék a tesztelésre.

**Pekny Márk:** Az első félévhez képest 180 fokot fordult minden. Az elején nehezen kezdtünk bele és nehéz volt rávenni magam a tanulásra, aztán mire észbe kaptunk már alig volt hátra 4-5 hónapunk a munka befejezésére. A zárómunkánkba belefektetett rengeteg idő meghozta az eredményét. Ebben a pár hónapban többet tanultam, mint az elmúlt években bármikor. Az alapos megtervezés után kemény munkával de elértük azt, hogy úgy nézzen ki a weboldalunk, ahogyan mi azt szeretnénk és még rengeteg további ötletem van, hogy hogyan tudnánk még tovább fejleszteni és szándékunkban is áll a vizsga után még foglalkozni vele. Legtöbbet a weboldal frontend részével bajlódtunk, mert ott voltak a legnehezebb kihívások, mivel React-ban csináltuk így rengeteg új ismerettel gazdagodtunk, de végül, előbb vagy utóbb sikerült leküzdenünk minden akadályt. Röviden és tömören elégedett vagyok magammal és a csapattal is és nagyon örülök, hogy idáig eljutottunk. Nagyon élveztük az egész ide vezető utat.

**Rapcsák Marcell:** A projekt során rengeteg új dolgot tanultam. Rengeteg nehézséggel néztem szemben, ugyanis ez volt az első komolyabb projektem amin csapatban dolgoztunk. A csapat összehangolása volt először a legnehezebb, ugyanis furcsa volt hogy nem egyedül dolgozunk és át kellett elemezni a többiek kódját. A kommunikáció hiánya viszont sosem lépett fel és mindig képesek voltunk tartani a kapcsolatot és meg tudtuk beszélni a dolgokat. Kicsit későn sikerült elkezdeni a projektet amiért hirtelen úgy tűnt hogy idő szűkében leszünk, viszont a csapatmunkával nagyon egyszerűen meg tudtunk oldani dolgokat ami jelentősen növelte a projekt előrehaladását. Összefoglalva nagyon élveztem a projektel töltött időt és rengeteg új dolgot tanultam és tanítottunk egymásnak.

### Kitűzött célok elérése

* **Edzésnapló és ranglista**: Az edzésnapló és a ranglista funkciók teljes mértékben működnek, a ranglistát modern dizájnnal és szűrési lehetőségekkel bővítettük.
* **Hírek szekció**: A hírek betöltése és magyarra fordítása sikeresen megvalósult, de a képek betöltése kezdetben problémát okozott. A proxy nélküli megoldás visszaállítása és a hibakezelés megoldotta a problémát.
* **Reszponzív dizájn**: Az oldal reszponzív lett, minden eszközön jól működik, de a kisebb képernyőkön a táblázatok görgethetősége további finomhangolást igényelhet.

### Felmerült problémák és megoldásuk

* **Képek betöltése**: A News API mezőjének betöltése CORS problémák miatt nem működött. Kezdetben proxy-t használtunk, de a felhasználói kérés alapján visszaállítottuk a közvetlen betöltést, és placeholder képekkel biztosítottuk a hibakezelést.
* **Világos/sötét mód eltávolítása**: A téma váltás logikáját el kellett távolítani, ami érintette az App.jsx, Home.jsx és hirek.css fájlokat. A folyamat során biztosítottuk, hogy a dizájn konzisztens maradjon.
* **Ranglista teljesítménye**: A ranglista oldalon a valós idejű frissítés túl gyakori volt, ezért a polling intervallumot 60 másodpercre növeltük.

### Saját fejlődés

A projekt során rengeteget tanultunk mind technikai, mind csapatmunkában szerzett készségek terén:

* **Technikai készségek**: Megtanultuk a React állapotkezelését, a REST API-k integrálását (News API, Google Translate API), valamint a reszponzív dizájn készítését Tailwind CSS segítségével. A MySQL adatbázis kezelését és az SQL lekérdezések optimalizálását is gyakoroltuk.
* **Problémamegoldás**: A képek betöltésével kapcsolatos problémák megoldása során megtanultuk, hogyan kell hibakeresést végezni (pl. böngésző fejlesztői eszközeivel), és hogyan kell alternatív megoldásokat találni (pl. placeholder képek használata).
* **Csapatmunka**: A projekt során szorosan együttműködtünk, és megtanultuk, hogyan osszuk fel a feladatokat (pl. Rapcsák a backend és adatbázis(dokumentáció) fejlesztésére,Pekny Márk a frontend dizájnra,PPT,(dokumentáció) Mózer Richárd Frontend elkészítése és az API integrációkra koncentrált illetve a dokumentáció ami közösen készült). A kommunikáció és a GitHub használata jelentősen javította a csapatmunkánkat és az otthoni munkát pedig a discord segítette.

Összességében a projekt során szerzett tapasztalatok megerősítették a programozási készségeinket, és felkészítettek minket a jövőbeli kihívásokra. Büszkék vagyunk arra, hogy egy működő, felhasználóbarát platformot hoztunk létre, amely valódi értéket nyújt a küzdősportok közösségének.

## Továbbfejlesztési lehetőségek

### Nem megvalósult ötletek

* **Valós idejű csevegés**: Terveztük egy chat funkció implementálását, amely lehetővé tette volna a felhasználók közötti valós idejű kommunikációt (pl. Socket.IO használatával). Az időkorlátok miatt ez nem valósult meg.
* **Értesítési rendszer**: Szerettünk volna egy értesítési rendszert, amely e-mailben vagy a felületen értesíti a felhasználókat az új edzésekről vagy eseményekről. Ez a funkció szintén kimaradt.

### Jövőbeli ötletek

* **Felhasználói kommentek a hírekhez**: A hírek szekcióban a felhasználók kommentelhetnének, ami növelné a közösségi interakciót.
* **Edzések részletesebb statisztikái**: Az edzésnaplóban részletesebb statisztikákat lehetne megjeleníteni (pl. havi bontás, edzéstípusok szerinti eloszlás).
* **Többnyelvű támogatás**: A platformot több nyelvre is le lehetne fordítani, hogy nemzetközi közönséget is elérjen.
* **Mobilalkalmazás**: A webalkalmazásból egy mobilalkalmazást lehetne készíteni React Native segítségével, hogy a felhasználók könnyebben hozzáférjenek a platformhoz.

# Felhasznált irodalom

 **Weboldalak**:

* React dokumentáció: <https://reactjs.org/docs/getting-started.html> (utoljára megtekintve: 2025.03.14.).
* Tailwind CSS dokumentáció: <https://tailwindcss.com/docs> (utoljára megtekintve: 2025.03.14.).
* News API dokumentáció: <https://newsapi.org/docs> (utoljára megtekintve: 2025.03.14.).
* Google Translate API dokumentáció: <https://cloud.google.com/translate/docs> (utoljára megtekintve: 2025.03.14.).

 **Elektronikus dokumentum**:

* Node.js hivatalos dokumentáció: <https://nodejs.org/dist/latest-v18.x/docs/api/> (letöltve: 2025.03.10.).
* AI (Chatgpt)

# Ábrajegyzék

[1. kép Ez a módosító ablak a programban 5](#_Toc63250394)