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).

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

\*\*5.3. A program telepítése (Fejlesztői környezet)\*\* 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://nodejs.org/)). Ellenőrizd a telepítést: `node -v` és `npm -v`. 2. \*\*MySQL:\*\* Telepítsd a MySQL 8.x vagy újabb verzióját ([https://dev.mysql.com/downloads/mysql/](https://dev.mysql.com/downloads/mysql/)). Jegyezd meg/állítsd be a `root` jelszót. 3. \*\*Git:\*\* Telepítsd a Git-et ([https://git-scm.com/](https://git-scm.com/)). Ellenőrizd: `git --version`. 4. \*\*(Opcionális) MySQL Workbench:\*\* Grafikus MySQL kliens ([https://dev.mysql.com/downloads/workbench/](https://dev.mysql.com/downloads/workbench/)). 5. \*\*(Opcionális) GitHub Desktop:\*\* Grafikus Git kliens ([https://desktop.github.com/](https://desktop.github.com/)). \*\*Telepítési lépések:\*\* 1. \*\*Projekt letöltése (Klónozás):\*\* \* Nyiss egy parancssort vagy terminált. \* Navigálj abba a könyvtárba, ahová a projektet telepíteni szeretnéd. \* Futtasd a következő parancsot (a megadott GitHub link alapján): ```bash git clone https://github.com/zaromunka\_kuzdosportok/React/KuzdoKert.git ``` \* Lépj be a letöltött `KuzdoKert` mappába: ```bash cd KuzdoKert ``` 2. \*\*Adatbázis beállítása:\*\* \* Győződj meg róla, hogy a MySQL szerver fut. \* Jelentkezz be a MySQL-be (pl. MySQL Workbench vagy parancssor: `mysql -u root -p`). \* Hozd létre az adatbázist: ```sql CREATE SCHEMA IF NOT EXISTS `kuzdosportok` DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4\_hungarian\_ci; ``` \* Importáld a projektben található `josql8.sql` fájl tartalmát az újonnan létrehozott `kuzdosportok` adatbázisba. MySQL Workbenchben: `Server` -> `Data Import` -> `Import from Self-Contained File`, válaszd ki a `josql8.sql`-t, és a `Default Target Schema`-nál válaszd ki a `kuzdosportok`-ot, majd `Start Import`. Parancssorból (a `KuzdoKert` mappából): ```bash mysql -u root -p kuzdosportok < josql8.sql ``` \*(Add meg a jelszót, ha kér.)\* \* \*\*Fontos:\*\* Ellenőrizd, hogy a `notifications` táblához hozzáadtad-e a `read TINYINT(1) NOT NULL DEFAULT 0` mezőt, mivel ez a frontend kódhoz szükséges, de hiányzott a megadott SQL sémából. Ha nem, add hozzá: ```sql USE kuzdosportok; ALTER TABLE `notifications` ADD COLUMN `read` TINYINT(1) NOT NULL DEFAULT 0 AFTER `message`; ``` 3. \*\*Backend függőségek telepítése és indítása:\*\* \* Navigálj a `Backend` mappába: ```bash cd Backend ``` \* Telepítsd a szükséges Node.js csomagokat: ```bash npm install ``` \* Indítsd el a backend szervert (általában fejlesztői módban): ```bash npm run dev ``` \*(Figyeld a konzolt, hogy a szerver sikeresen elindult-e, és melyik porton figyel, pl. 5000)\* 4. \*\*Frontend függőségek telepítése és indítása:\*\* \* Nyiss egy \*új\* parancssort/terminált. \* Navigálj a `Frontend` mappába: ```bash cd ../Frontend ``` \*(Vagy a teljes elérési úttal a `KuzdoKert` mappából)\* \* Telepítsd a szükséges Node.js csomagokat: ```bash npm install ``` \* Indítsd el a Vite fejlesztői szervert: ```bash npm run dev ``` \*(Figyeld a konzolt, hogy a frontend alkalmazás melyik címen érhető el, pl. http://localhost:5173)\* 5. \*\*Alkalmazás elérése:\*\* \* Nyisd meg egy modern webböngészőt. \* Írd be a címsorba a frontend fejlesztői szerver címét (pl. `http://localhost:5173`). \* Az alkalmazásnak most már futnia kell a helyi gépeden. \*\*5.4. A program használatának részletes leírása\*\* Ez az útmutató lépésről lépésre bemutatja a "Küzdő Kert" platform funkcióinak használatát.

## 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

# 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

Fel kell sorolnod az összes olyan szoftver eszközt amelyet a program fejlesztéséhez, illetve a dokumentáció készítéséhez felhasználtál. (Pl. programozási nyelv, fejlesztői környezet, adatbázis-kezelő rendszer, kép-, szöveg-, zeneszerkesztő program stb.)

* Ha használtál mások által kifejlesztett modulokat, akkor azt is le kell írnod, a forrás pontos megjelölésével.
* Tájékozódj ezeknek a moduloknak a jogszerű felhasználásáról!

Ajánlott terjedelem: ½ -1 oldal.

## Adatmodell leírása

A feladatban alkalmazott adatbázis vagy adatszerkezet részletes leírása

* Adatbázis esetén az adattáblák leírása, a közöttük lévő kapcsolatok megadása, lehetőleg diagram is legyen
* Ha nincs adatbázis, akkor a program fő adatszerkezeteinek a specifikációja, célszerű diagramot is rajzolni
* OOP jellegű megvalósítás esetében az UML osztálydiagram

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

## Részletes feladatspecifikáció, algoritmusok

A program lényeges függvényeinek, az osztályok metódusainak a specifikációja (mit valósít meg az adott függvény, illetve metódus, milyen paraméterei vannak, mi a visszatérési érték)  Az algoritmizálható részek leírása valamilyen algoritmus-leíró eszközzel (struktogram, pszeudo-kód, esetleg UML aktivitás-diagram) Ajánlott terjedelem: a feladat jellegétől függően 2-5 oldal. 5. Forráskód  A teljes forráskódot a nyomtatott dokumentációba nem kell beletenni!  Lehet viszont a nyomtatott dokumentációban a fontosabb kódrészeket magyarázattal szerepeltetni

## 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:**

### 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)