Debreceni Egyetem

Informatikai Kar

Szakdolgozat

# Fullstack labdarúgó tippjáték készítés Spring Boot és React keretrendszerekkel

## Témavezető: Készítette:

## Major Sándor Roland Szarvas Péter

## Tanársegéd Programtervező informatikus

Debrecen

2025

# Bevezetés

Az informatika és a játéktervezés rohamos fejlődése az elmúlt évtizedekben alapjaiban változtatta meg a szórakoztatóipart, lehetőséget teremtve a klasszikus játékok digitalizálására és kreatív újragondolására. A sportfogadás és a tippelős játékok sem maradtak ki ebből a folyamatból, hiszen a modern technológiák – például adatbázisok, backend és frontend keretrendszerek – segítségével új szintre emelhetők ezek az élmények. Számos weboldal létezik ma már, amely sporteseményekhez kapcsolódó tippelési lehetőségeket kínál (pl. TippmixPRO [1]), ahol a felhasználók valós időben követhetik az eredményeket és versenghetnek egymással.

Szakdolgozatom témájaként egy focimeccs-eredmény tippelős weboldal megvalósítását választottam, amely nem csupán a hagyományos tippjátékok digitális adaptációja, hanem egyedi funkciókkal bővített, interaktív élményt nyújtó platform. A projektem egy olyan online játékot hoz létre, amely ötvözi a focimeccsek kiszámíthatatlan izgalmát és a stratégiai gondolkodást, miközben a modern webfejlesztési technológiák erejét kihasználva közösségi és kompetitív elemekkel gazdagítja a játékot.

A „GoalRush” nevű játékom legfontosabb újítása a pontozási és ranglista-rendszer. A felhasználók a valós focimeccsek eredményeire tippelhetnek, és pontokat szerezhetnek az alábbiak szerint: egy tökéletesen eltalált eredmény (gólszám és nyertes csapat) 3 pontot ér, a nyertes csapat helyes megtippelése (vagy döntetlennél a döntetlen ténye, de eltérő gólszámmal) 1 pontot ad, minden egyéb esetben pedig 0 pont jár. A játék ingyenes, a pontok csupán a versengést és a ranglétrákon való előrehaladást szolgálják. A rendszer heti és havi ranglétrákat kínál, így a játékosok különböző időtávokon mérhetik össze tudásukat és szerencséjüket. A felhasználók regisztrációval saját profilt hozhatnak létre, amelyen követhetik teljesítményüket és eredményeiket.

A projekt relevanciája több szempontból is kiemelkedő. Egyrészt a focirajongók széles közönségét célozza meg, miközben a tippelés stratégiai mélysége gondolkodásra és elemzésre ösztönzi a játékosokat – mindezt ingyen, kockázat nélkül tehetik meg, szemben a legtöbb sportfogadási oldallal, ahol pénz vagy digitális kredit szükséges egy tipp leadásához. A fiatalabb generáció számára, akik a videójátékok dinamizmusához szoktak, a ranglétrák és a közösségi versengés izgalmasabbá teszi az élményt. Másrészt fejlesztési szempontból is jelentős, hiszen bemutatja, hogyan lehet valós idejű adatokat integrálni egy interaktív játékba modern technológiák – például a MongoDB adatbázis, a Spring Boot backend és a React frontend – felhasználásával. Harmadrészt a projekt demonstrálja ezeknek az eszközöknek az alkalmasságát skálázható, felhasználóbarát webalkalmazások készítésére.

A témaválasztásom személyes érdeklődésemből fakad: gyermekkorom óta rajongok a fociért, és mindig is kíváncsi voltam, hogyan lehet digitális platformokon keresztül új élményeket teremteni. A konkrét ötletet a témavezetőmmel közösen pontosítottuk, így döntöttem a Java alapú Spring Boot keretrendszer, a MongoDB NoSQL adatbázis és a React frontend kombinációja mellett. A MongoDB rugalmassága lehetővé tette a felhasználói adatok és meccseredmények hatékony kezelését, a Spring Boot stabil backendet biztosított, a React pedig modern, reszponzív felületet kínált a játékosok számára.

A jövőben szeretném továbbfejleszteni a projektet. Terveim között szerepel egy valós idejű értesítési rendszer bevezetése, amely figyelmezteti a játékosokat a közelgő meccsekre, valamint egy baráti liga funkció, ahol a felhasználók saját csoportokat hozhatnak létre. Emellett a pontozási rendszert is bővíteném, például bónuszpontokkal a különösen nehéz meccsek helyes megtippeléséért. A felhasználói felületet is tovább csiszolnám, mivel a fejlesztés során a backendre helyeztem a nagyobb hangsúlyt, így a frontend terén látok még lehetőséget a finomításra. Hosszabb távon az adatszerzést is optimalizálnám, például a külső API-k helyett saját webscraper megoldással, hogy még nagyobb kontrollom legyen az adatok felett. A fejlesztési folyamat során felmerült kihívások megoldása sok tanulást és kreativitást igényelt, amit a jövőben tovább kamatoztatnék a projekt tökéletesítésére.

# Tárgyalás

## Felhasznált eszközök

## Git / GitHub

A projekt verziókezelő rendszerének a Git-et [2] választottam, ez egy nyílt forráskódú, Linus Torvalds által létrehozott verziókezelő rendszer (VCS). A Linux kernel fejlesztéséhez hozta létre Linus 2005-ben. Ez a rendszer lehetővé tette számára a kódváltoztatásainak a nyomon követését, a verziók kezelését.

A verziókezelő rendszerek használata manapság már-már kijelenthető, hogy kötelező a fejlesztői projekteknél, mivel akár vissza lehet térni egy esetleges fennálló hiba esetén egy korábbi verzióhoz, akár csapatban, akár egyedül több ágon folyhat a fejlesztés, amelyeket később egyesíteni lehet gyorsan és hatékonyan. Lehetőség van verziók kiadására, akár alfa teszt, béta teszt vagy teljes kiadás, a kiadás is lehet főbb verziószámú vagy apróbb hibajavító csomag.

A forráskódom GitHub-on [3] elérhető (<https://github.com/Sz4rv4s/SZAKDOLGOZAT>), ez a Microsoft felhő alapú Git szolgáltatása, ami nem csak tárhelyként funkcionál, hanem hibajegyek létrehozására, kezelésére, kollaborációra, CI/CD-re egyaránt használható.

### Docker, GCP

A Docker konténerizációs platformot használtam a fejlesztés során a MongoDB adatbázisom, valamint az ahhoz tartozó Express nevű webes felület létrehozására, mivel a fejlesztés során több számítógépen dolgoztam és szükség volt a konzisztens, egyszerű telepítésre, átalakításra, törlésre, mind fejlesztési, mind tesztelési szempontból.

A Docker Compose nevű parancsablaki eszköz segítségével egy YAML leírófájlból készítettem azonos konténereket.

## MongoDB NoSQL adatbázisrendszer

## Java és Spring Boot keretrendszer

## Spring Dotenv, Lombok

## TypeScript és React

## Tailwind CSS, Axios, Zustand, Formik, Yup

## Kivitelezési folyamatok

### Az adatbázis és az adatok tárolásának kivitelezése

### A felhasználókezelés és időzítések kivitelezése

### A REST szervizek és végpontok létrehozása

### A frontend elkészítése

### A tesztek és naplózások hozzáadása

### Az elkészült projekt publikálása Google Cloud Platformon keresztül

## Használati útmutató

# Összefoglalás

# Irodalomjegyzék