

**SAPIENTIA ERDÉLYI MAGYAR TUDOMÁNYEGYETEM
MAROSVÁSÁRHELYI KAR,
INFORMATIKA SZAK**



SAPIENTIA
ERDÉLYI MAGYAR
TUDOMÁNYEGYETEM

Tanuljunk programozni és teszteljük tudásunk online tesztekkel

DIPLOMADOLGOZAT

Témavezető:
Dr.Osztián Erika,
Adjunktus

Végzős hallgató:
Horvath János

2023

UNIVERSITATEA SAPIENTIA DIN CLUJ-NAPOCA
FACULTATEA DE ȘTIINȚE TEHNICE ȘI UMANISTE,
SPECIALIZAREA INFORMATICĂ



UNIVERSITATEA
SAPIENTIA

Învățăm să programăm și să testăm cunoștințele noastre folosind
teste online

LUCRARE DE DIPLOMĂ

Coordonator științific:
Dr.Osztián Erika,
Lector universitar

Absolvent:
Horvath János

2023

**SAPIENTIA HUNGARIAN UNIVERSITY OF
TRANSYLVANIA
FACULTY OF TECHNICAL AND HUMAN SCIENCES
COMPUTER SCIENCE SPECIALIZATION**



SAPIENTIA
HUNGARIAN UNIVERSITY
OF TRANSYLVANIA

Learn programming and test our knowledge with online tests

BACHELOR THESIS

Scientific advisor:
Dr.Osztián Erika,
Phd. lecturer

Student:
Horvath János

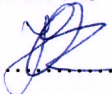
2023

Declarație

Subsemnatul/a HORVATH JÁNOS....., absolvent(ă) al/a specializării INFORMATICA....., promoția 2023... cunoscând prevederile Legii Educației Naționale 1/2011 și a Codului de etică și deontologie profesională a Universității Sapientia cu privire la furt intelectual declar pe propria răspundere că prezenta lucrare de licență/proiect de diplomă/disertație se bazează pe activitatea personală, cercetarea/proiectarea este efectuată de mine, informațiile și datele preluate din literatura de specialitate sunt citate în mod corespunzător.

Localitatea, TÂRGU MUREȘ,
Data: 2023.06.14

Absolvent

Semnătura.....

LUCRARE DE DIPLOMĂ

Coordonator științific:
dr. Osztián Erika

Candidat: **Horváth János**
Anul absolvirii: 2023

a) Tema lucrării de licență: Învățăm să programăm și să testăm cunoștințele noastre folosind teste online

b) Problemele principale tratate:

- Metode de învățare a programării
- Testele online ca instrument de evaluare
- Dezvoltarea și utilizarea platformelor online de învățare și testare
- Evaluarea rezultatelor obținute prin teste online

c) Desene obligatorii:

- Diagrama arhitecturii sistemului, inclusiv interacțiunea dintre aplicație API și interfața profesorului
- Capturi de ecran al interfeței aplicației și interfeței pagina de web

d) Softuri obligatorii:

- Aplicația mobilă comunică cu o bază de date printr-o API pentru a extrage informații, întrebări și rezultate relevante. Scopul aplicației este de a oferi un mediu interactiv de învățare și practică pentru cei interesați de programare.
- Site-ul web permite administratorului să gestioneze baza de date, întrebările și datele utilizatorilor. Administratorul site-lui poate efectua operații ca adăugarea, actualizarea și ștergerea informațiilor din baza de date.

e) Bibliografia recomandată:

- [1] Flutter's Documentation (<https://docs.flutter.dev/>)
- [2] React.js Documentation (<https://legacy.reactjs.org/docs/getting-started.html>)
- [3] React.js library Documentation (<https://ant.design/docs/react/introduce>)

f) Termene obligatorii de consultații: de 2-3 ori pe lună

g) Locul și durata practicii: Universitatea „Sapientia” din Cluj-Napoca,
Facultatea de Științe Tehnice și Umaniste din Târgu Mureș, laboratorul 414

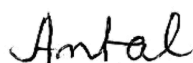
Primit tema la data de: 15.10.2022

Termen de predare: 15.06.2023

Semnătura Director Departament



Semnătura responsabilului
programului de studiu



Semnătura coordonatorului



Semnătura candidatului



Kivonat

Dolgozatom témája egy olyan telefonos alkalmazás fejlesztése ami megkönnyíti a programozás világába belépők számára a tanulást, alapok elsajátítását és állandó gyakorlást biztosít különböző tesztek segítségével. Az alkalmazás mellé társul még egy weboldal amely segítségével hozhatunk létre új teszteket, kategóriákat szerkeszthetjük azokat és nyomon követhetjük a felhasználók eredményeit.

Az alkalmazás és a weboldal létrehozásához különböző technológiákat használtam, dolgozatom rendelkezik még egy API-al amin keresztül kommunikál egyaránt a weboldal és az alkalmazás a közös adatbázissal. Az alkalmazás tanítási módja teszt/kvízes megközelítés mellett esett mivel, a napjainkban rohanó világban az okostelefonunk mindig egy kéznyújtásnyira van tőlünk és így bármikor lehetőségünk van a gyakorlásra amihez elég lehet pár percet időnkől rászánunk az alkalmazás használatára. A teszt/kvízes megközelítés egy szórakoztató, játékos tanulást biztosít, amely egy kellemes felhasználói élményt eredményez és így eredménye-sebbé teheti a tanulást. Az adminisztrátor/adminisztrátorok folyamatosan bővíthetik az applikáció tanítási lehetőségeit kérdések, kategóriák hozzáadásával a weboldalon keresztül, emellett nyomon követhetik a felhasználók tevékenységeit, és alapvető adminisztratív feladatokat tudnak ellátni.

Az elkészült alkalmazás hasznos lehet az egyetemi hallgatók számára, főleg az első éves hallgatók, vagy a felvételizők számára, akik csak most ismerkednek az informatika világával.

Továbbiakban a dolgozatomat fogom részletesen bemutatni, az alkalmazás megtervezésének lépéseit, működési elvét. Ezen kívül szó lesz a felhasznált technológiákról, elért célokról és jövőbeli tervekről.

Rezumat

Tema lucrării mele este dezvoltarea unei aplicații mobile care facilitează învățarea și asimilarea noțiunilor de bază în lumea programării și oferă o practică constantă prin intermediul diferitelor teste. În plus, aplicația este însoțită de un site web care permite crearea de noi teste, editarea categoriilor și monitorizarea rezultatelor utilizatorilor.

Pentru crearea aplicației și a site-ului web, am folosit diverse tehnologii. Lucrarea mea include și o interfață de programare a aplicațiilor (API), prin intermediul căreia site-ul și aplicația comunică cu o bază de date comună. Modalitatea de învățare a fost abordată prin teste și quiz-uri, deoarece în lumea agitată în care trăim, smartphone-ul este întotdeauna la îndemână și ne oferă posibilitatea de a exersa chiar și în câteva minute de timp disponibil. Abordarea prin teste și quiz-uri oferă o învățare distractivă și interactivă, rezultând o experiență plăcută pentru utilizatori și contribuind la eficientizarea procesului de învățare. Administratorul/administratorii pot extinde în mod constant posibilitățile de învățare ale aplicației prin adăugarea de întrebări și categorii prin intermediul site-ului web, pot monitoriza activitățile utilizatorilor și pot îndeplini sarcini administrative de bază.

Aplicația finalizată poate fi utilă studenților universitari, în special celor din primul an sau celor care se pregătesc pentru admitere și se familiarizează cu lumea informaticii.

În continuare, voi prezenta în detaliu lucrarea mea, pașii de proiectare ai aplicației, principiul funcționării și tehnologiile utilizate, precum și obiectivele atinse și planurile viitoare.

Abstract

The topic of my thesis is the development of a mobile application that facilitates learning and mastering the basics of programming for beginners, while providing constant practice through various tests. The application is accompanied by a website where users can create new tests, edit categories, and track their results.

To create the application and website, I utilized various technologies. My thesis also includes an API that enables communication between the website and the application, using a shared database. The learning approach in the application is based on tests and quizzes, as in today's fast-paced world, our smartphones are always within reach, allowing us to practice with just a few minutes of our time. The test-based learning approach provides an enjoyable and interactive learning experience, making learning more effective. Administrators have the ability to continuously expand the learning opportunities of the application by adding questions and categories through the website. They can also monitor user activities and perform basic administrative tasks.

The completed application can be beneficial for university students, especially first-year students or those preparing for entrance exams who are just beginning to explore the world of computer science.

Furthermore, I will present my thesis in detail, including the steps involved in designing the application, its functioning principle, the technologies used, the achieved goals, and future plans.

Tartalomjegyzék

1. Bevezető	11
2. Célkitűzések	12
3. Elméleti megalapozás	13
4. Követelmény specifikáció	15
4.1. Telefonos alkalmazás követelményei	16
4.1.1. Felhasználói követelmények	16
4.1.2. Rendszer követelmények	17
4.2. Adminisztrációs weboldal követelményei	18
4.2.1. Felhasználói követelmények	18
4.2.2. Rendszer követelmények	19
5. Tervezés	20
5.1. Használt technológiák	20
5.1.1. Flutter	20
5.1.2. ReactJs	22
5.1.3. NodeJs	22
5.2. Fejlesztés menedzselése	24
5.2.1. Felhasználói felület megtervezése	24
5.2.2. Feladatok beosztása	24
5.2.3. Adatbázis	24
5.3. Szekvencia diagrammok	27
6. A szoftver bemutatása	33
6.1. Telefonos alkalmazás	33
6.1.1. Bejelentkezés oldal	33
6.1.2. Regisztráció oldal	33
6.1.3. Menü oldal	33
6.1.4. Egyszemélyes játékmód	35
6.1.5. Gyors kvíz játékmód	35
6.1.6. Profil	35
6.1.7. Beállítások	38
6.1.8. Statisztikák	38
6.2. Adminisztrációs weboldal	39
6.2.1. Bejelentkezés	39

6.2.2.	Fő oldal	39
6.2.3.	Adminisztrátorok oldal	39
6.2.4.	Felhasználók oldal	39
6.2.5.	Kérdések oldal	39
6.2.6.	Tantárgyak és fejezetek oldal	39
6.2.7.	Statisztika oldal	40
6.2.8.	Gyors kvíz oldal	40
6.3.	API	40
6.4.	Tovább fejlesztési lehetőségek	41
Összefoglaló		43
Ábrák jegyzéke		45

1. fejezet

Bevezető

Az informatika világa számos izgalmas területet foglal magába, és a programozás az egyik legfontosabb és alapvető eleme ennek a területnek. A programozás az informatika egyik alapvető készsége, amely lehetővé teszi a számítógépekkel való kommunikációt és az alkalmazások, szoftverek létrehozását.

A programozás során a programozók különböző programozási nyelveket használnak, amelyek segítségével utasításokat adnak a számítógépnek. Ezen sorok után valami bonyolult dolognak tűnhet a programozás és épp ezért lett a dolgozatom témája egy olyan típusú alkalmazás fejlesztése, amely segít kezdőknek betekintést nyerni ebbe a világba, folyamatos gyakorlással fejlődést biztosítani.

Dolgozatom középpontjában egy telefonos alkalmazás áll, ami egyaránt működik IOS és Android operációs rendszerrel rendelkező okostelefonokon. Az applikáció egy tanító, gyakorló jellegű telefonos alkalmazás, amely egy API-al kommunikál az adatok elérése érdekében, itt az adatok alatt gondolok a tesztekben szereplő kérdésekre, személyes adatokra, eredményekre. A tesztek személyre szabottak, egy adminisztrátor által, épp ezért kerülhetnek be könnyebb és nehezebb kérdések, témák az adatbázisba. Az applikáció a tesztelés, kvízes jellegű megközelítéssel próbál tanulási lehetőséget biztosítani az érdeklődők számára, ez a megközelítés interaktív tanulást biztosít. Az okostelefon elterjedésével pedig könnyedén elérhető a fejlődési lehetőség, hiszen bármikor és bárhol előkapva okostelefonunk, könnyedén gyakorolhatunk. A kvízek egy másik előnye, hogy azonnali visszajelzést kapunk eredményünkről, a tesztek végén láthatjuk mit rontottunk el, és következő alkalomkor már tudni fogjuk, hogy ezt a hibát ne kövessük el, hiszen a hibáinkból tanulhatunk a legtöbbet.

A személyre szabott témák és tesztek segítségével, könnyedén letudjuk takarni bármely témát a programozás tág világából. Ezt a személyre szabást egy adminisztrátor/adminisztrátoroknak tervezett weboldal segítségével valósíthatjuk meg. A weboldal segítségével az adminisztrátor a kérdések és témák bővítése mellett, a felhasználók adatait is tudja módosítani, ami hasznos lehet ha a felhasználónk elfelejti a belépési jelszavát vagy ha módosítani, vagy akár törölni akarjuk a felhasználót a rendszerből.

Az alkalmazás rendelkezik statisztikákkal, vagyis vissza tudjuk nézni megoldott tesztjeinket, hogy teljesítettünk rajtuk, melyik kérdésre mit válaszoltunk és hogy a válaszuk helyes volt-e vagy sem. Az eredményeinket a weboldalon is megtudjuk nézni, minden tesztre.

2. fejezet

Célkitűzések

A legfőbb célkitűzésem egy olyan alkalmazás létrehozása volt ,amellyel képesek vagyunk betekintést nyújtani olyan programozási alapokba az érdeklődők számára ,amely megkönnyíti majd fejlődésük folyamatát és folyamatos gyakorlási lehetőséget biztosít a megtanultak elmélyítésére ,emellett pedig a könnyű és gyorsan frissíthető adatbázis kezelés megteremtése ,mivel így az applikáció szolgáltatásai nem lesznek korlátozottak.

A dolgozatom egy nagyon fontos alapköve egy adatbázis lett ,amelyben eltudom tárolni a felhasználók adatait, a tesztekhez szükséges adatokat, és a teszteken szerzett adatokat. A saját munkánk és tanulásunk elkülönítése érdekében fontosnak láttam elkülöníteni adatainkat, ezért szükséges volt kialakítani egy autentikációs rendszert, így saját eredményeinkhez csak mi férünk hozzá és a saját ritmusunkban haladhatunk.

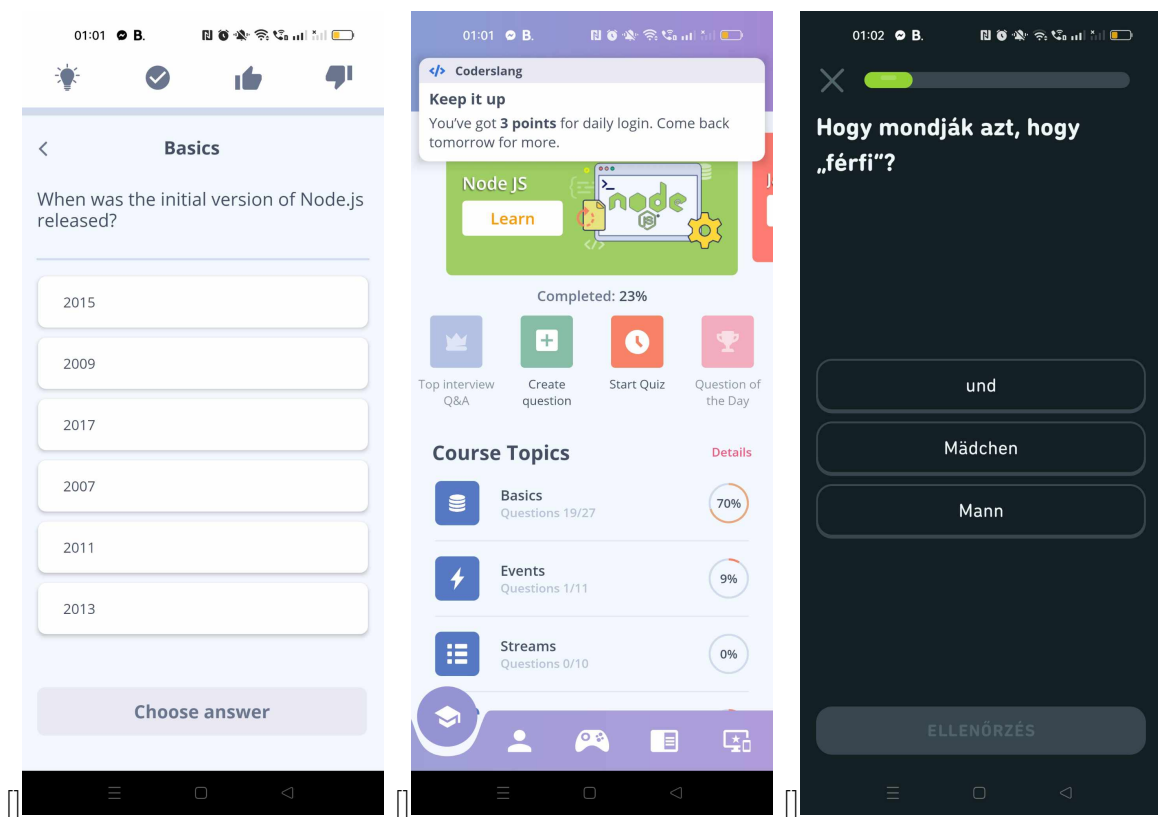
Nem utolsó sorban elengedhetetlennek láttam egy web oldal kifejlesztését ,ahonnan kezelhetjük az adatbázisunkat, témákat, kérdéseket adhatunk hozzá, módosíthatunk, egyaránt a felhasználók adatait is, és nyomon követhetjük eredményeiket.

3. fejezet

Elméleti megalapozás

A projekt elkezdése előtt sokáig tűnődtem, hogy mi is lenne a legjobb megközelítés, hogy egyszerű és letisztult de hasznos alkalmazást tudjak létrehozni, ami egy kellemes felhasználói élményt is tud nyújtani.

Elsődleges célom az volt, hogy az egyszerűség mellett sokoldalú legyen az alkalmazás és ekkor tudtam hogy szükségem lesz egy megfelelő adminisztrációs weboldalra, ahol bármikor tudom frissíteni a kérdéseim, témáim adatbázisát. Elsődleges inspirációm a **Duolingo** alkalmazás volt, innen tetszett meg az egész játékos tanulás ötlet, utána próbáltam keresni hasonló alkalmazást aminek kicsit több köze volt az informatikához és megtaláltam a **Coderslang** nevű alkalmazást, amelyben csak programozási nyelvekkel kapcsolatos kvízek találhatók. Az utolsó alkalmazás, amit az egyetemen ismertem meg, **Kahoot** adott inspirációt, innen jött a gyors kvíz ötlet, amellyel egy adminisztrátor létrehozhat egy kvízt egy adott intervallumra, adott felhasználóknak.

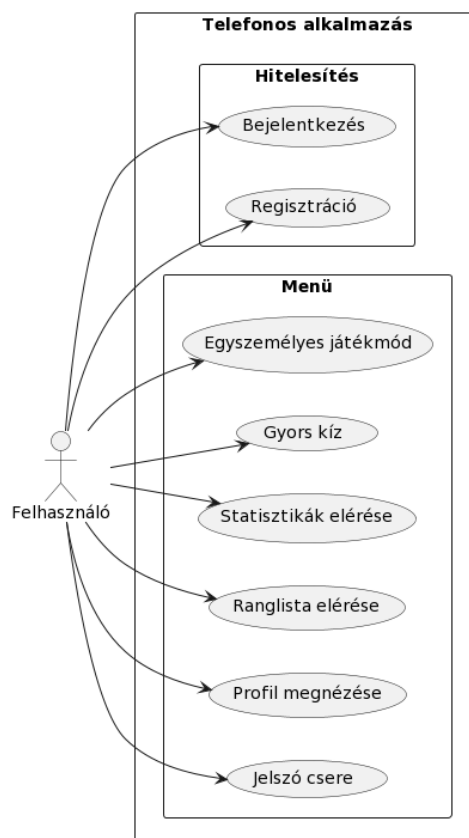


3.1. ábra. Hasonló alkalmazások

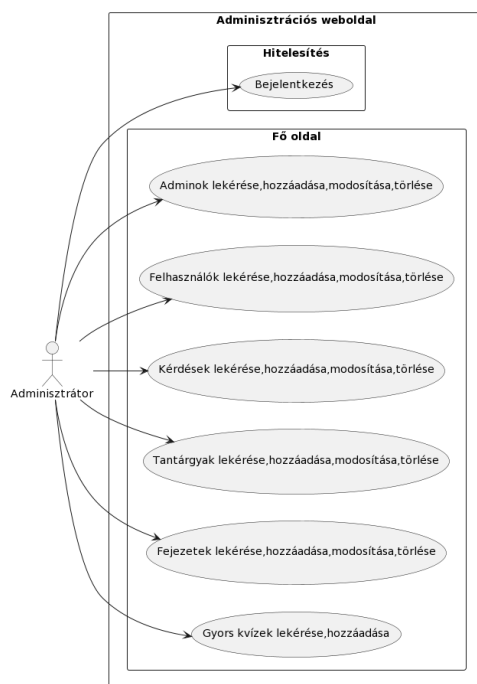
4. fejezet

Követelmény specifikáció

Az alkalmazás gördülékeny működésének érdekében, létezik néhány követelmény melyeknek a felhasználó és a rendszer eleget kell hogy tegyen. A fejezet további részében ezeket a követelményekről fogok beszélni. Mivel a dolgozatom tartalmaz egy weboldalt és egy telefonos alkalmazást, ezért a mindkettőhöz kapcsolódó követelményeket külön-külön fogom tárgyalni. A 4.1 diagramon láthatjuk tanulóknak esetében a telefonos alkalmazás funkcióit amiket a felhasználók igénybe vehetnek, míg a 4.2 ábra az adminisztrációs weboldalról mutatja be ugyanezt.



4.1. ábra. Használati eset diagram (tanulók esetén)



4.2. ábra. Használati eset diagram
(adminisztrátorok esetén)

4.1. Telefonos alkalmazás követelményei

A dolgozatom középpontjában a telefonos alkalmazás áll, ezért is úgy vélem ezzel kell kezdenem a követelmények bemutatását.

4.1.1. Felhasználói követelmények

Magától értetődően a legfontosabb követelmény hogy a felhasználó rendelkezzen egy okostelefonnal, amely egyaránt lehet Android operációs rendszerű vagy akár IOS, ha ezek birtokában vagyunk akkor sikeresen elindíthatjuk az alkalmazást, de sajnos ez a használatához még nem elég, mivel szükségünk van internet kapcsolatra is, hogy tudjunk bejelentkezni az alkalmazásba és annak szolgáltatásait tudjuk igénybe venni.

Abban az esetben ha a felhasználó még nem rendelkezik fiókkal, regisztrációval orvosolhatja ezt a problémát és utána pedig bejelentkezhet a frissen létrehozott fiókjával. A regisztrációhoz szükséges adatok a következők: vezetéknév, keresztnév, felhasználó név, email cím, jelszó, nem (férfi/nő), iskolai felkészültség (itt választhat beépített opciók közül) és születési dátum.

Miután megtörtént a sikeres bejelentkezés, a felhasználó igénybe veheti az alkalmazás szolgáltatásait, bejelentkezés után az alkalmazás fő képernyőjére kerülünk, a menübe ahol kiválaszthatjuk, hogy mit szeretnénk csinálni, legfelső opciókként választhatunk a játék lehetőségek között.

Választhatunk egyszemélyes játékmódot, amelyben kiválasztunk egy tantárgyat és abból egy fejezetet, majd azokból kapunk kérdéseket, amelyekre válaszolnunk kell, játék

kezdetén van 5 életünk és minden hibás válasz után elveszítünk egy életet, ha elveszítjük az összes életünket akkor a játéknak vége és a kvíz sikertelennek bizonyult, ha egymás után 3 kérdésre helyesen válaszolunk nyerhetünk egy életet, de életeink száma így sem haladhatja meg az 5-öt, ha elfogynak életeink az sem probléma mivel minden 2 perc elteltével visszakapunk egy életet. Ha sikeresen elvégzünk egy kvízt, ami azt jelenti hogy nem fogytak el életeink, akkor a kvíz végén megtekinthetjük az eredményeinket, és hogy melyék kérdésre mit választottunk.

Egy másik opció lehet a gyors kvíz opció, ahol csak akkor van lehetőségünk játszani, ha egy adminisztrátor létrehoz egy gyors kvízt az adminisztrációs weboldalas felületből, és a megadott időkorlaton belül használjuk. A játék menet itt is pontosan ugyanúgy zajlik, annyi eltéréssel hogy itt nem rendelkezünk életekkel.

A játék lehetőségeken kívül, a felhasználó megnézheti a profilját, megváltoztathatja a jelszavát, a játék nehézségét, ami azt jelenti hogy így minden témából és fejezetből csak olyan szintű kérdéseket fog kapni, ez a gyors kvíz opcióra nem vonatkozik. Ezeket az opciókat a Profil és Beállítások fülek alatt találjuk meg. Ezek az opciók még fejlesztés alatt állnak, nem működnek teljesen jól de már tartalmazza őket az alkalmazás.

Nem utolsó sorban, a felhasználó megtekintheti az eredményeit, az elvégzett tesztek eredményeit, s hogy melyek mit választott az adott kérdésekre. Ezek mind a Statisztika fül alatt találhatóak, itt meg találunk egy ranglistát is, ahol láthatjuk, hogy különböző felhasználóknak milyen eredményeik, átlaguk van az alkalmazásban.

4.1.2. Rendszer követelmények

A telefonos alkalmazás működéséhez, egyaránt használhatunk Android és IOS operációs rendszerrel rendelkező okostelefont, az alkalmazásunk nem igényel magas rendszer követelményt, egy Android 4.1 vagy újabb típusú operációs rendszer gond nélkül megkéne oldja a problémáinkat, IOS típusú okostelefonokon is hasonlóan elvárható, hogy nagyjából az összes okostelefon képes az alkalmazásunk működtetésére, az alacsony rendszer követelmények miatt.

Az okostelefonunknak mindenképp rendelkeznie kell internet hozzáféréssel, és 200-300 MB tárhellyel.

Funkcionális követelmények

- Felhasználói regisztráció és bejelentkezés
- Kvíz játék
- Kategóriák és nehézségi szintek választása
- Rangsorolás
- Statisztikák megjelenítése
- Felhasználó adatainak megjelenítése
- Felhasználói jelszó csere

Nem funkcionális követelmények

- Aktív internet kapcsolat
- Android vagy IOS operációs rendszerrel rendelkező okostelefon

4.2. Adminisztrációs weboldal követelményei

Az adminisztrációs weboldal azzal a céllal készült, hogy a telefonos alkalmazás adatait folyamatosan tudjuk frissíteni, új témákkal, fejezetekkel, kérdésekkel. Mindezeken túl, szerkeszthetjük a felhasználók adatait, kitörölhetjük őket és nyomon tudjuk követni eredményeiket.

4.2.1. Felhasználói követelmények

A weboldal működéséhez elengedhetetlen egy böngésző és egy aktív internet kapcsolattal rendelkező eszköz.

Ha rendelkezünk az előbb említett feltételekkel akkor már csak egy regisztrált fiókra lesz szükségünk, amit mi magunktól nem hozhatunk létre, mivel a weboldal úgy lett kigondolva, hogy rendelkezik egy adminisztrátorral, akit az adatbázisban adunk hozzá tervezés elején és ő majd adhat hozzá más adminisztrátorokat.

Ha megtörtént a sikeres bejelentkezés, akkor a weboldal tovább irányít minket az alap oldalra, ahol a nem rég elvégzett teszteket láthatjuk. A weboldal bal oldalán található egy kis menü, ahonnan elérhetjük az összes oldalt, és igénybe vehető szolgáltatást.

Az első opció a menüből az az „Adminok” fül, amellyel szerkeszteni tudjuk a jelenlegi adminisztrátorokat, hozzá adni újat, vagy akár törölni őket. Minden adminisztrátorról a következő információkat jelenítjük meg egy táblázatban: keresztnév, vezetéknév és email cím. Ha hozzá szeretnénk adni új adminisztrátort, akkor az előbb említett információkat kell megadnunk, és egy jelszót.

A következő fül a „Felhasználók” fül, ahol az előző fülhöz hasonlóan egy táblázatban megjelenítjük a felhasználókat és azok adatait, itt már több adatunk van, mégpedig a következők: keresztnév, vezetéknév, felhasználó név, email cím, születési dátum, nem, tanulmányok. A felhasználók adatai szerkeszthetők, hozzá is adhatunk új felhasználót és törölhetünk is meglévő felhasználót. Egy új felhasználó hozzáadásához az előbb említett információkat kell megadnunk, és egy jelszót.

A „Kérdések” fül alatt jeleníthetők meg a meglévő kérdések, itt tudunk bármikor hozzáadni kérdéseket, szerkeszteni a meglévő kérdéseket, keresni közöttük és törölni őket. A kérdésekről a következő adatokat jelenítjük meg a táblázatban: kérdés, válaszok, helyes válasz/helyes válaszok, tantárgy, fejezet és nehézség, ha új kérdést szeretnénk létrehozni akkor ezeket az információkat kell megadnunk miután az új kérdés gombra kattintottunk és kitöltöttük a kívánt adatokat.

A „Tantárgyak és Fejezetek” fül alatt szerkeszthetjük a meglévő tantárgyakat, fejezeteket, hozzá adhatunk újakat, egy fejezetet átrakhatunk más tantárgyhoz, törölhetünk, fontos megjegyezni egy fejezet, tantárgy törlése magával vonja az összes hozzá tartozó kérdés törlését.

A „Statisztika” fül ahogy a neve is árulkodik, az elvégzett tesztekéről szolgáltat adatot, itt kereshetünk az elvégzett tesztek között, több paraméter szerint is.

Az utolsó fül amit tartalmaz a kis baloldali menü, a „Gyors Kvíz”, amely által létrehozhatunk egy tesztet, amit kiadhatunk az adatbázisból választott kérdésekkel, egy adott idő intervallumban, és ezalatt az idő lehet csak megoldani.

4.2.2. Rendszer követelmények

Az adminisztrációs weboldal működéséhez elengedhetetlen az aktív internet kapcsolat, és egy böngésző, a weboldal bármely operációs rendszeren fut, de asztali számítógépekre volt tervezve, ezért azon ajánlott használni és nem telefonokon, mivel ott nem volt tesztelve.

Funkcionális követelmények

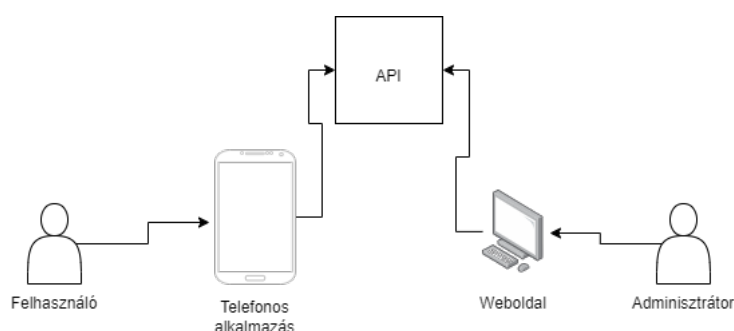
- Felhasználói bejelentkezés
- Adatok gyors és átlatható megjelenítése
- Adatok szűrése és keresése
- Adatok bevezetése az adatbázisba (kérdések, tantárgyak, fejezetek stb)
- Statisztikák megjelenítése
- Felhasználó adatainak megjelenítése
- Felhasználói jelszó csere
- Felhasználók létrehozása

Nem funkcionális követelmények

- Aktív internet kapcsolat
- Internetes böngésző (Google Chrome, Mozilla Firefox, Microsoft Edge stb.)

5. fejezet

Tervezés



5.1. ábra. A rendszer architektúrája

A 5.1 ábrán látható szoftver egy Flutter telefonos alkalmazás és egy ReactJS-ben készült adminisztrációs weboldal funkcióit implementálja. Mindkét alkalmazás egy általam készített API-al kommunikál az adatok elérése érdekében, ez az összeköttetés az adatbázis és a két alkalmazás között. Az API Node.js-ben készült, továbbiakban az általam felhasznált technológiákról fogok beszélni.

5.1. Használt technológiák

5.1.1. Flutter

A Flutter egy keresztplatformos keretrendszer, amelyet a Google fejlesztett ki, nagy erőssége, hogy egy kódbázisban fejleszthető egyidejűleg Android és iOS alkalmazás. A keretrendszer a Dart programozási nyelvet használja.

Flutter előnyei:

- **Keresztplatformos fejlesztés:**Flutter lehetővé teszi a keresztplatformos fejlesztést, ami azt jelenti hogy, ugyanabban a kódbázisban fejleszthetünk egyszerre Android és iOS alkalmazást, ami jelentős időmegtakarítást eredményezhet
- **Gyors és kiváló teljesítmény:**A keretrendszer platformfüggetlen felhasználói felületet biztosít, és ezáltal majdnem natív sebességű és teljesítményű alkalmazásokat

hozhatunk létre. A beépített Widgetek, a hardveres gyorsítás és a JIT (Just-In-Time) fordítás hozzájárulnak a gyors és sima felhasználói élményhez.

- **Gazdag funkcionalitás:**A keretrendszer, rengeteg beépített funkciót és bővítést áll rendelkezésünkre. támogatja az animációkat, a képek manipulálását, a hálózati kommunikációt, a helymeghatározást, a platformspecifikus API-k elérését és még sok más.
- **Hot Reload:**A Hot Reload funkció lehetővé teszi a fejlesztők számára, hogy valós időben lássák a változtatásokat a kódban, anélkül, hogy újra kellene indítaniuk az alkalmazást. Ez a gyors visszajelzési ciklus lehetővé teszi a gyors prototípus készítést és az alkalmazás iteratív fejlesztését.

Flutter hátrányai:

- **Méret:**A Flutter alkalmazások mérete nagyobb lehet, mint egy azonos alkalmazás natív fejlesztéssel. Ez annak köszönhető, hogy a Flutter a saját felhasználói felületi keretrendszerét és motorját használja, amelyek hozzájárulnak a méretnövekedéshez.
- **Teljesítményproblémák az összetett UI-k esetén:** Bár a Flutter általában nagyon gyors és hatékony, összetettebb felhasználói felületek esetén, például nagy mennyiségű animációval vagy nagy adatmennyiséggel, előfordulhat némi teljesítményromlás.

Dart előnyei:

- **Könnyen tanulható és olvasható:**Könnyen tanulható, mivel számos hasonlóság figyelhető meg a Dart és más ismert nyelvek között ilyen például a Java és a JavaScript, emellett tiszta és olvasható szintaxissal rendelkezik, amely jelentősen megkönnyíti a vele való munkát.
- **Hatékony és gyors:**Dartot a magas teljesítmény és a hatékonyság jellemzi. A nyelv JIT (Just-In-Time) és AOT (Ahead-of-Time) fordítást is támogat, ami lehetővé teszi a gyors indítást és futtatást. Emellett a Dart optimalizált futási környezetet és memóriakezelést kínál, ami hozzájárul a kiváló teljesítményhez.
- **Modern nyelvi funkciók:**A Dart programozási nyelv számos modern nyelvi funkcióval rendelkezik, támogat, ilyen például az erős típusellenőrzés, aszinkron programozás, generikus típusokkal és lambda kifejezésekkel.

Dart hátrányai:

- **Kisebbségi fejlesztői közösség:**Annak ellenére hogy a Flutter és a Dart folyamatosan ismertebb és ismertebb lesz, még mindig elég kicsinek mondható az elterjedtsége, ismertsége más népszerűbb programozási nyelvekhez képest. Ezt azt eredményezi, hogy kevesebb dokumentációt és oktató anyagot találunk hozzá.

- **Korlátozott támogatás más platformokon:** Dartot a magas teljesítmény és a hatékonyság jellemzi. A nyelv JIT (Just-In-Time) és AOT (Ahead-of-Time) fordítást is támogat, ami lehetővé teszi a gyors indítást és futtatást. Emellett a Dart optimalizált futási környezetet és memóriakezelést kínál, ami hozzájárul a kiváló teljesítményhez.
- **Korlátozott ökoszisztéma:** Bár a Flutter könyvtárai széles körűek és fejlett funkciókkal rendelkeznek, a Dart nyelv maga nem rendelkezik olyan gazdag és fejlett ökoszisztémával, mint más nyelvek, például a JavaScript vagy a Python. Ez azt jelenti, hogy kevesebb harmadik féltől származó könyvtár, modul és eszköz érhető el a fejlesztési folyamat támogatására.

5.1.2. ReactJs

A ReactJS egy egyre felkapottabb JavaScript könyvtár, amely Facebook fejlesztői által lett létrehozva, a felhasználói felületek fejlesztésére használják.

ReactJs előnyei:

- **Komponens alapú fejlesztés:** A ReactJS komponens alapú architektúrát használ, amely lehetővé teszi a fejlesztők számára a felhasználói felület könnyű és moduláris felépítését. A komponensek önálló UI darabok, amelyeket újra felhasználhatunk, így nem ismételjük a már megírt kódot és gyorsabban haladhatunk a fejlesztésben.
- **Virtuális DOM:** A ReactJS a virtuális DOM-ot használja a hatékony és gyors felhasználói felület frissítésére. A virtuális DOM egy absztrakció a tényleges DOM felett, amely lehetővé teszi a hatékony újrarajzolást és a felhasználói interakciókra történő gyors válaszadást.
- **Deklaratív megközelítés:** A ReactJS deklaratív megközelítést alkalmaz a felhasználói felület fejlesztésében. Ez azt jelenti, hogy a fejlesztők csak kifejezik, hogy milyen legyen a felhasználói felület, és a ReactJS gondoskodik az állapot és a felületi frissítések automatikus kezeléséről.

ReactJs hátrányai:

- **Időigényes tanulás:** A ReactJS egyedi megközelítéssel rendelkezik a felhasználói felületek területén, ami azt jelenti, hogy akik a hagyományos webfejlesztéshez vannak hozzá szokva, kicsit furcsa és időigényes lehet számukra ez a fajta megközelítés.
- **Komplexitás növekedése:** Bár a komponens alapú fejlesztés lehetővé teszi a kód moduláris szerkezetét, a nagyobb és bonyolultabb alkalmazások esetén a komponensek hierarchiája és az adatáramlás kezelése bonyolultabbá válhat.

5.1.3. NodeJs

Node.js egy szerveroldali JavaScript futtatókörnyezet, amely lehetővé teszi a JavaScript használatát a szerveroldali fejlesztés során. Míg a hagyományos böngészőalapú

JavaScript a kliensoldali fejlesztésre szolgál, a Node.js lehetővé teszi a JavaScript futtatását a szerveren.

NodeJs előnyei:

- **Gyors és hatékony:** Gyorsaságát a Google V8 motorjának köszönheti, mivel ez nagyon hatékonyan kezeli a JavaScript kódot, ami lehetővé teszi a gyors és hatékony fejlesztést.
- **Aszinkron működés:** Az aszinkron működés miatt hasznos lehet, főleg API-ok esetén, mivel több kérést tud egyszerre kezelni, mivel nem blokkolják egymást a folyamatok, események. Az aszinkron működése miatt nagyon jól skálázható és hatékony.
- **Nagy modulválaszték:** Nagy előnye, hogy hatalmas modulválasztékkal rendelkezik, ami nagyon hasznos lehet, mivel rendelkezésünkre áll rengeteg csomag, amiket nem kell folyamatosan újra írunk, így rengeteg időt spórolhatunk meg hiszen csak a Node Package Manager nevű csomagkezelővel fel kell telepítenünk őket.
- **Egységes nyelv:** A Node.js lehetővé teszi a JavaScript használatát mind a kliensoldalon, mind a szerveroldalon. Ez egységesíti a fejlesztési környezetet, mivel ugyanazt a nyelvet használhatjuk a teljes alkalmazásfejlesztési folyamat során.

NodeJs hátrányai:

- **Egyszálas működés:** Az egyszálas működés azt jelenti hogy a Node.js egy fő szálon futtatja a kódot, ami az aszinkron működés ellenére is, megterhelő lehet (pl. pl. intenzív CPU-használatot igénylő számítás esetén). Ezért fontos az aszinkron műveletek megfelelő kezelése, hogy elkerüljük a fő szál blokkolódását.
- **Memóriahasználat:** A Node.js az alkalmazások futtatásához a memóriát használja, ez azt jelenti hogy a hosszú folyamatoknál (pl. a háttérben futó feladatok esetén) vagy a nagy memórafogyasztású alkalmazásoknál a memóriaerőforrások hatékony kezelése fontos lehet.
- **Kevésbé alkalmas CPU-intenzív műveletekre:** A Node.js erőssége inkább az I/O-intenzív alkalmazásokban és a hálózati kommunikációban rejlik. Mivel a JavaScript egy szálon fut, a CPU-intenzív műveletek (pl. hosszú távú számítások) lassabbak lehetnek a Node.js-ben. Ilyen esetekben érdemes lehet más technológiák vagy nyelvek (pl. Python vagy Java) használata.

5.2. Fejlesztés menedzselése

Dolgozatom során igyekeztem kisebb nagyobb sikerrel arra törekedni, hogy a munkám menetrendje a cégekéhez hasonlóan történjen, ezért tervezgettem, jegyeteket készítettem, mielőtt nekiláttam volna valaminek.

5.2.1. Felhasználói felület megtervezése

Miután eldöntöttem milyen irányba is szeretnék menni az alkalmazással, következő lépésként elkezdtem tervezni az alkalmazás keretvázát, kinézetét amit a Figma weboldal segítségével oldottam meg, habár a tervezés után nagyon sokat változott az alkalmazás kinézete, ez segített elindítani és elképzelni az egész alkalmazás kezdetleges változatát a fejemben.

5.2.2. Feladatok beosztása

A dolgozatom írása közben, próbáltam felosztani feladataimat és egyszerre csak egy konkrét dologra koncentrálni és lépésről lépésre haladni, erre pedig a Trello weboldalt használtam ami nagyon hasznosnak tűnt az én esetemben mivel mindig tudtam követni hogy jelenleg melyék feladat épp hol tart, van-e befejezve vagy kellene elkezdni.

5.2.3. Adatbázis

A telefonos alkalmazáshoz és a weboldalhoz ugyanazt az adatbázist használtam, mégpedig egy MySql adatbázist, ezt tűnt nekem a legkézenfekvőbbnek, mivel ebben van a legtöbb ismeretem adatbázisok terén és úgy láttam ebben tudom legjobban összekapcsolni az adatokat az alkalmazásomhoz.

Az adatbázis tervezésekor először is szükségem volt felhasználókra, ezt a célt szolgálja a **users** tábla, amelyben a mezők a következő tulajdonságokra vonatkoznak:

- **firstName:** felhasználó vezetékneve
- **lastName:** felhasználó keresztnéve
- **userName:** személyre szabott felhasználónév
- **password:** jelszó
- **email:** email cím
- **status:** ez alapján dönti el a rendszer hogy rendes felhasználó vagy adminisztrátor az illető
- **gender:** a felhasználó neme
- **education:** a felhasználó iskolai végzettsége, felkészültsége
- **birthdate:** felhasználó születésnapja

- **isdeleted:** törölve van-e vagy sem a rendszerből, ezt a törlési módot választottam mivel, így nem végzünk végleges törlést és visszalehet állítani a törlési adatokat

Ebben a táblában tároljuk az összes felhasználót, adminisztrátorokat és tanulókat egyaránt. A status mezővel teszünk különbséget köztük. Ha regisztrálunk új felhasználót akár weboldalról akár a telefonos alkalmazásból az ebbe a táblába fog bekerülni.

A **users** táblához kapcsolatban van 4 táblával, ezek a **lives**, **education**, **gender** és **quizresults** táblák.

A **lives** táblában tároljuk a felhasználók életeit, amik nagyon fontosak az egyik játékmódhoz, a tábla 3 mezővel rendelkezik egy azonosítóval (id), egy referenciával a felhasználóra, amely a felhasználó azonosítója (userId) és a az utolsó mező tárolja az életek számát (livesNumber). Fontos megjegyezni, hogy a livesNumber mező ha nem haladja meg a maximális értéket akkor minden percben növekedik 1-el.

A **gender** és **education** táblákban tároljuk a nem típusokat (férfi és nő), míg az education táblában a végzettség típusokat, mindkét tábla rendelkezik egy azonosítóval és egy elnevezéssel.

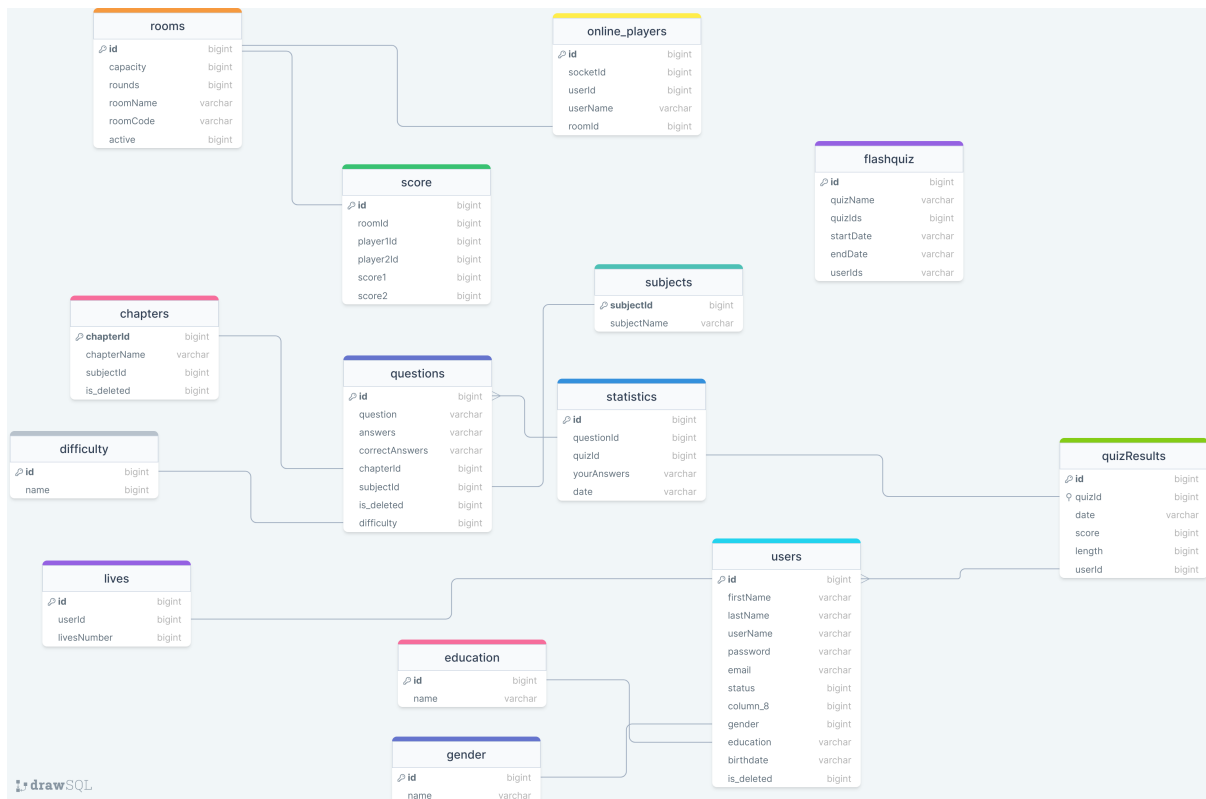
Az utolsó tábla ami kapcsolódik a users táblához a **quizresults** tábla, amelyben tároljuk a teszt eredményeket. A következő mezőkkel rendelkezik:

- **quizId:** minden teszt elkezdése előtt létrejön egy egyedi azonosító, így tudjuk a teszteket megkülönböztetni egymástól és lekérni a statistics táblából hogy melyék teszthez melyék kérdések tartoztak.
- **date:** a teszt elvégzésnek időpontja
- **score:** elért pontszám
- **length:** a teszt hossza, összes elérhető pontszám
- **userId:** a felhasználó azonosítója aki elvégezte a tesztet

A következő nagyon fontos tábla a **questions** tábla, itt tároljuk a kérdéseket, az adminisztrációs weboldalról ha beszurunk új kérdést ide fog bekerülni. Minden kérdés a következő mezőkkel rendelkezik:

- **question:** ez a mező tartalmazza a konkrét kérdést
- **answers:** ez a mező tartalmazza a válasz lehetőségeket, mindegyiket elválasztva egymástól egy #-el
- **correctAnswers:** itt tároljuk a helyes válaszokat, ugyanúgy egy összevont karakterláncban ahol egymástól egy #-el vannak elválasztva
- **chapterId:** ez a mező egy számot tartalmaz, ami a fejezet azonosítója, amelyhez tartozik a kérdés
- **subjectId:** a chapterId-hoz hasonlóan ez is egy szám és egy azonosító csak ez a tantárgy azonosítója

- **is_deleted:** ez a mező arra szolgál hogy el tudjuk dönteni, hogy aktív-e a kérdés az adatbázisban, lényegében így törölünk kérdést, ennek előnye hogy visszaállítható a kérdés ha meggondoljuk magunkat
- **difficulty:** ez is egy azonosító az előző kettőhöz hasonlóan, ez a kérdés nehézségi szintjét sorolja be egy kategóriába a difficulty tábla alapján



5.2. ábra. A rendszer adatbázisa

Ez a **question** tábla kapcsolatban áll a **chapters**, **subjects**, **statistics** és **difficulty** táblákkal.

A **chapters** és **subject** táblák nagyon hasonlóak, a **subject** táblákban tároljuk a tantárgyakat, minden tantárgyhoz tartozik több fejezet, ezeket a **chapters** táblákban tároljuk a **chapters** tábla rendelkezik egy elnevezéssel (**chapterName**), egy mezővel amivel tudjuk ellenőrizni hogy töröltük-e a fejezetet (**is_deleted**), fontos megjegyezni egy fejezet törlése magával vonja az összes hozzá tartozó kérdés törlését is, ezen kívül még rendelkezik egy mezővel ami tartalmaz egy azonosítót ami azt mutatja hogy melyék tantárgyhoz tartozik a fejezet.

A **subject** tábla rendelkezik egy mezővel ami az elnevezését tárolja, és hasonlóan a **chapters** táblához egy mezővel ami azt mutatja hogy töröltük-e a fejezetet az adatbázisból. Ha törölünk egy tantárgyat akkor töröljük az összes hozzá tartozó fejezetet és az összes hozzá tartozó kérdést.

A **difficulty** tábla tartalmaz egy elnevezést és egy alap azonosítót amit minden tábla tartalmaz, itt tároljuk a nehézségi szinteket, ezek a nehézségi szintek a kérdéseket sorolják csoportokba

A **statistics** tábla azt tárolja el, hogy minden kvíz/teszt alatt melyék kérdésre milyen választ adtunk és hogy mikor, egyik mezője tartalmaz egy azonosítót ami a kérdésre a mutat a kérdés táblából (questionId), a következő mező azt mutatja hogy melyék teszthez tartozik ez a válasz, ez is egy azonosító mivel minden teszthez generálunk egy egyedi azonosítót ahogy az előbb említettem (quizId). Elmaradhatatlan egy mező amibe elmentjük a mi általunk adott választ válaszokat (yourAnswers) és az utolsó mező amit tartalmaz ez a tábla a dátum amikor adtuk ezt a választ és létrejött ez a statisztika (date).

Az adatbázisunk itt még nem ér véget, mivel rendelkezik még pár táblával, amik közül az egyik a **flashquiz** nevű tábla, amely a gyors kvíz játék módban létre hozott teszteket tárolja, mégpedig a következő mezőkkel:

- **quizName:** az adminisztrátor által választott kvíz név
- **questionIds:** ebben tároljuk el a kiválasztott kérdések azonosítóit
- **startDate:** a teszt indítási dátuma és időpontja
- **endDate:** a teszt zárási dátuma és időpontja
- **userIds:** ebben tároljuk el a kiválasztott felhasználók azonosítóit, így csak számukra lesz elérhető a teszt

Még maradt három táblánk az adatbázisban, amik egy fejlesztésben lévő játékmód adataid készültek tárolni, ahol a **rooms** tábla tárolta volna a létrehozott szobát amit egyik felhasználó hozott volna létre és megosztotta volna egy másik felhasználóval a szoba adatait így az csatlakozott volna és egymás ellen mérhették volna fel tudásukat, az **online_players** tábla tárolta volna a játékosokat és a **score** az eredményt, a játékmód egy 80% százalékos készenléti állapotban van, de még sajnos nem teljesen funkcionális.

5.3. Szekvencia diagrammok

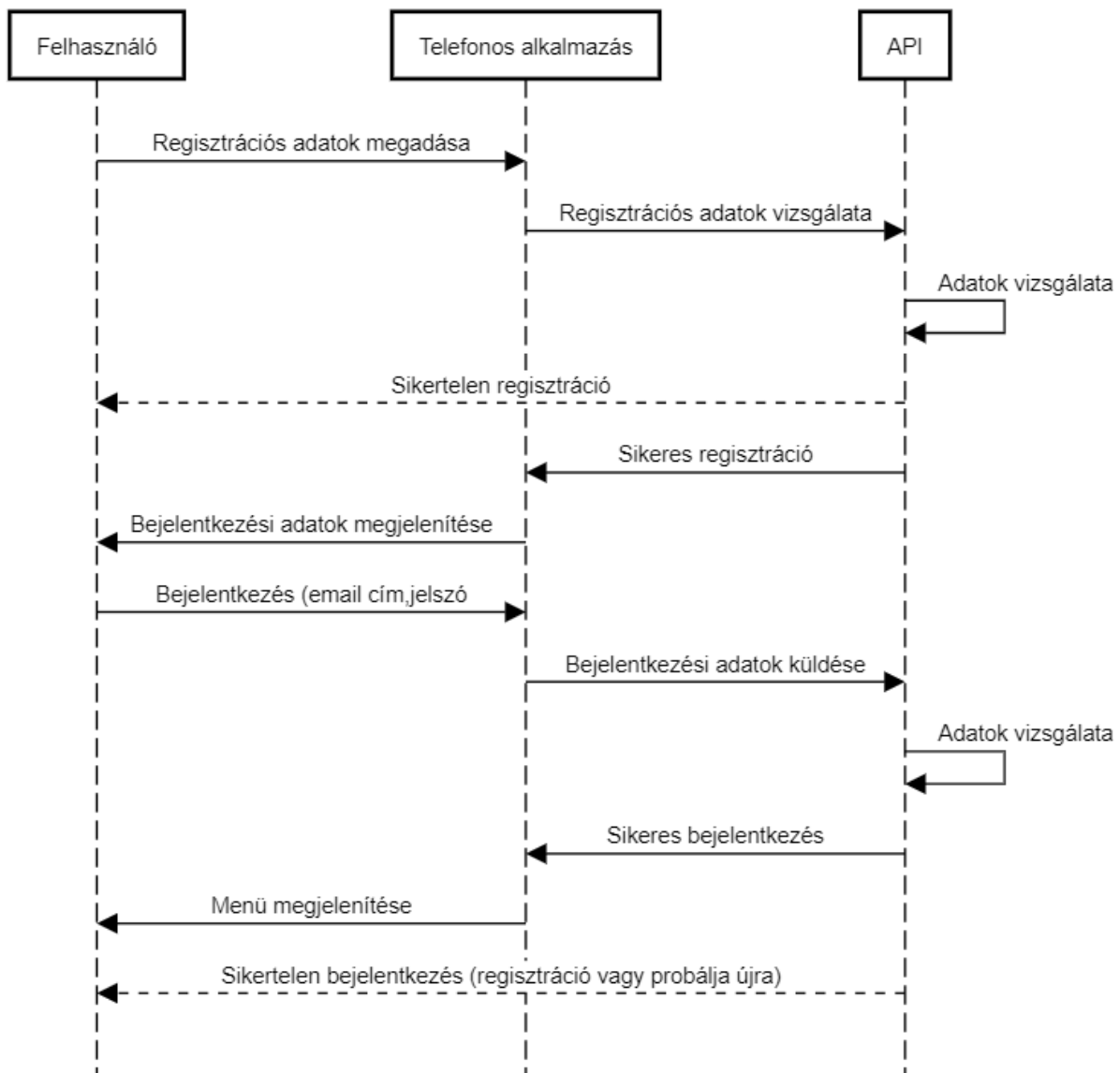
A 5.3 ábra bemutatja egy szekvencia diagram segítségével, a telefonos alkalmazás bejelentkezési és regisztrációs folyamatát. A felhasználó ha nem rendelkezik felhasználóval, kitölti a regisztrációs adatokat, ezt elküldi az API-nak egy POST kéréssel, az megpróbálja beszúrni azt az adatbázisba, sikeres beszúrás esetén, bejelentkezhetünk az újonnan létrehozott felhasználó fiókkal. Ha nem volt sikeres a regisztráció újra ki kell töltenünk a mezőket. Ha rendelkezünk már felhasználó fiókkal, akkor meg kell adnunk az email címünket és a jelszót, ez egy POST kérést fog küldeni az API-nak, sikeres bejelentkezés esetén az alkalmazás átirányít minket a menüre. Sikertelen bejelentkezés esetén hiba üzenetet kapunk.

Az 5.4 és 5.5 ábrán látható a játékmenetek szekvencia diagramjai, amelyek bemutatják miután kiválasztjuk a játékmódot, egy GET kéréssel lekérjük a kérdéseket, előkészíti az alkalmazás a háttérben a játék képernyőt majd indul is a kvíz, ahol megadjuk a választunkat, és „Következő” gombra kattintva navigálunk a következő kérdésre, a háttérben

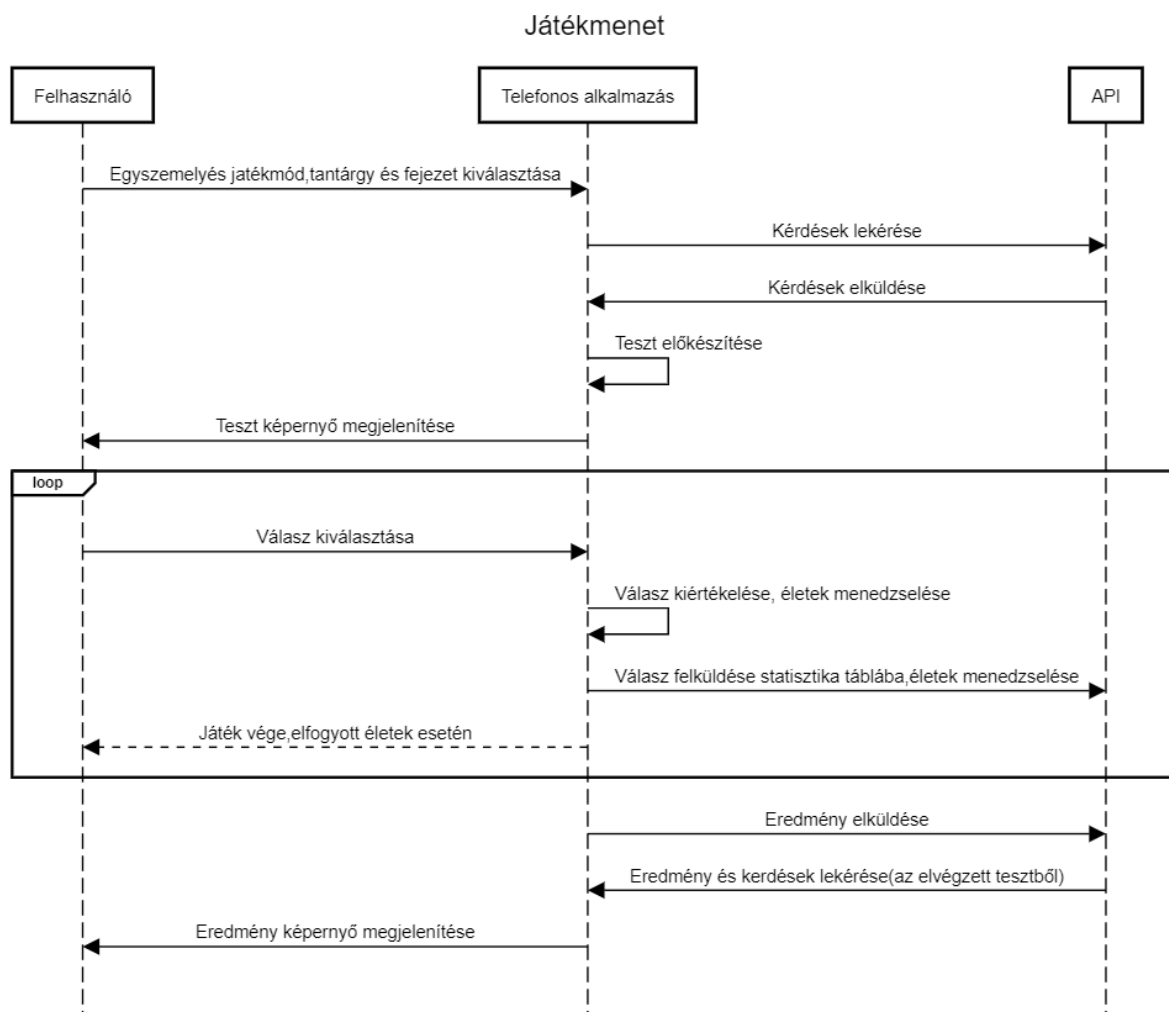
pedig egy POST kéréssel felküldjük a statisztika táblába a válaszunkat, az egyszemélyes játékmód esetében menedzseljük a felhasználó életeinek számát is (növeljük vagy kivonunk belőle). Ha az utolsó kérdéshez érünk, a „Következő” gomb átalakul „Teszt befejezése” gombbá és felküldjük az utolsó statisztikát és az eredményt az eredmények táblába egy újabb POST kéréssel. Az alkalmazás ezután átnavigál az eredmények oldalra ahol egy GET kéréssel lekérjük a kérdéseket és a válaszokat a nemrég elvégzett tesztre, erre azért van szükség hogy megtudjuk nézni mit rontottunk el és mire válaszoltunk helyesen. Fontos megjegyezni ha az egyszemélyes játékmódban elfogynak az életeink játék közben, a játék véget ér. A gyors kvíz játékmódban nincsenek életeink, nem kell ilyen problémával szembenéznünk.

A 5.6 ábrán látható, hogyan érjük el a telefonos alkalmazásban a statisztika és profil adatait, mindkét esetben amikor rákattintunk az oldalra, egy GET kérés kerül végrehajtásra, majd az API válaszként visszaküldi a kívánt adatokat és ezeket jeleníti meg az alkalmazás számunkra.

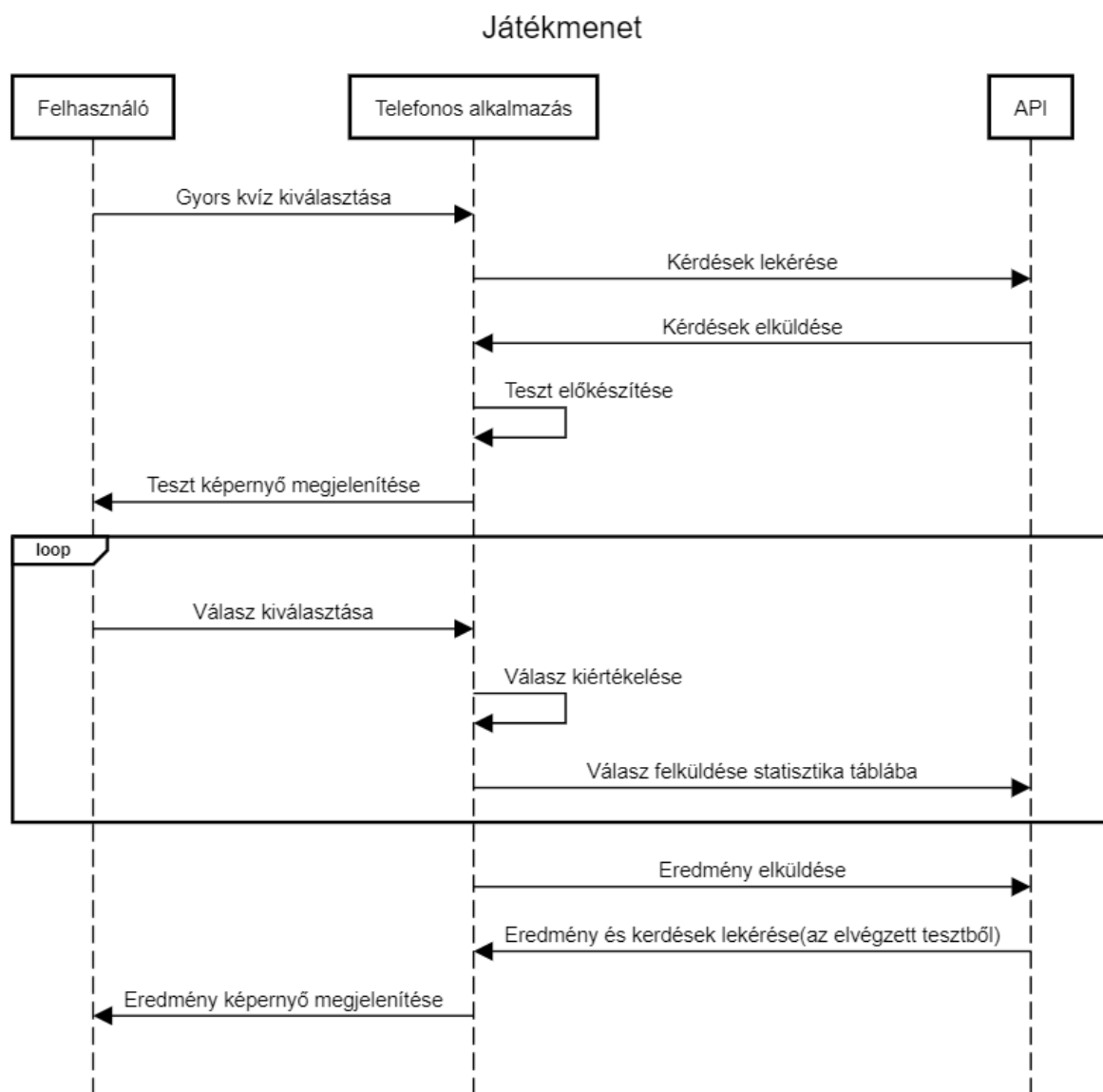
Bejelentkezés és Regisztráció



5.3. ábra. Telefonos alkalmazás bejelentkezés és regisztráció

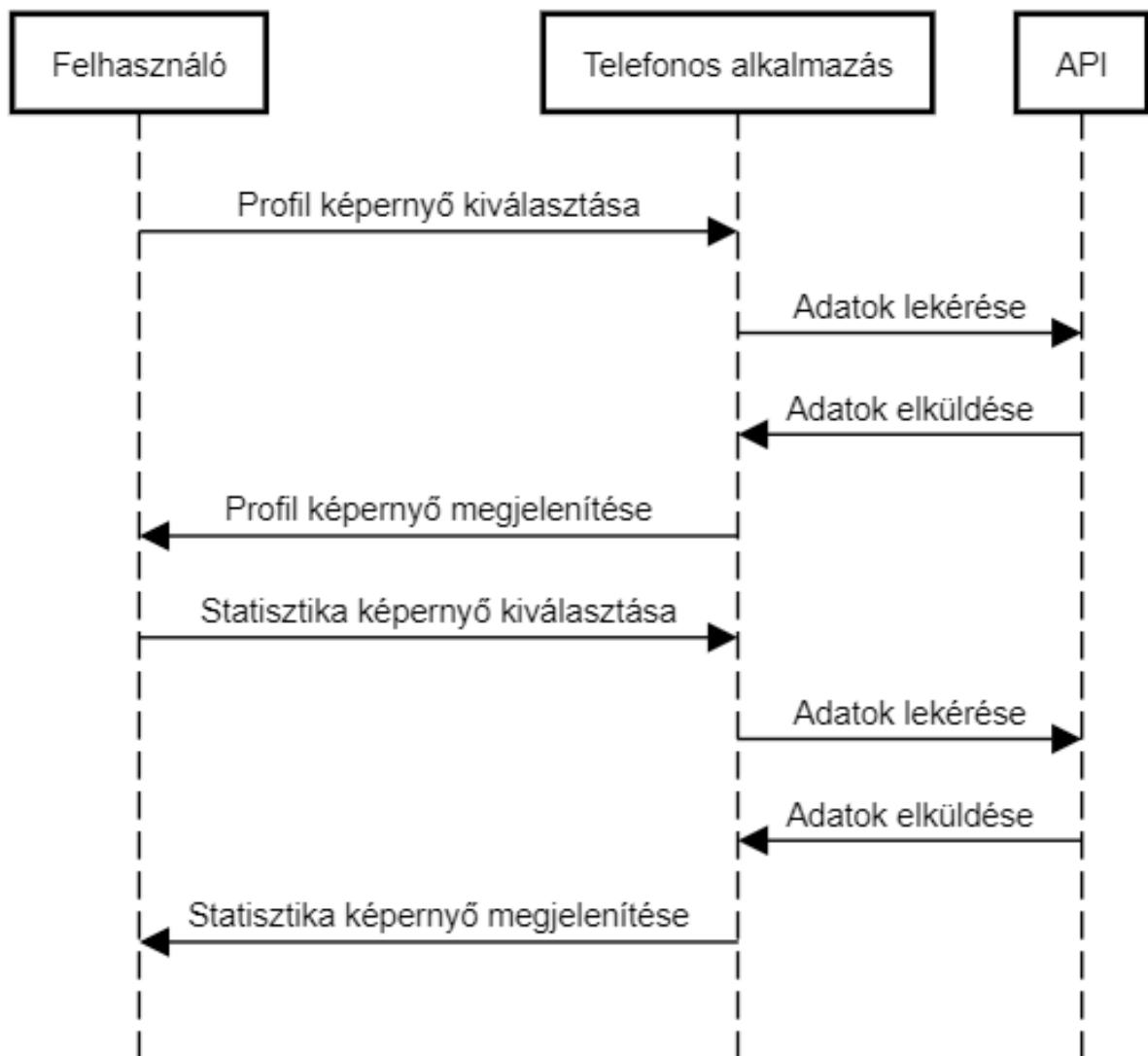


5.4. ábra. Egyszemélyes játék szekvencia diagram



5.5. ábra. Gyors kvíz szekvencia diagram

Statisztikák és profil lekérése



5.6. ábra. Statisztika és profil oldal szekvencia diagram

6. fejezet

A szoftver bemutatása

A következő fejezetben az általam fejlesztett szoftverről lesz szó, ami tartalmazni fog egy részletes bemutatást egyaránt a telefonos alkalmazásról és az adminisztrációs weboldalról emellett még szó lesz az API-ról amelyen keresztül kommunikálnak az adatbázissal.

6.1. Telefonos alkalmazás

6.1.1. Bejelentkezés oldal

A telefonos alkalmazás megnyitásakor az első képernyő a bejelentkezésért felel, itt meg kell adnunk egy email címet és egy jelszót amennyiben van regisztrált felhasználónk. A jelszó mező rendelkezik egy kis gombbal amely által megtudjuk nézni hogy helyesen írtuk-e a jelszót, mivel az alaptól beíráskor kivan pontozva mint minden alkalmazásnál biztonsági okok miatt. Ha netán elfelejtettük a jelszavunkat, megváltoztathatjuk az elfelejtett jelszavam linkre kattintva.

Ez a kezdőképernyő, [6.1](#) ábrán van szemléltetve, amennyiben nem rendelkezünk felhasználóval létrehozhatunk egyet a „Regisztrálj” linkre kattintva.

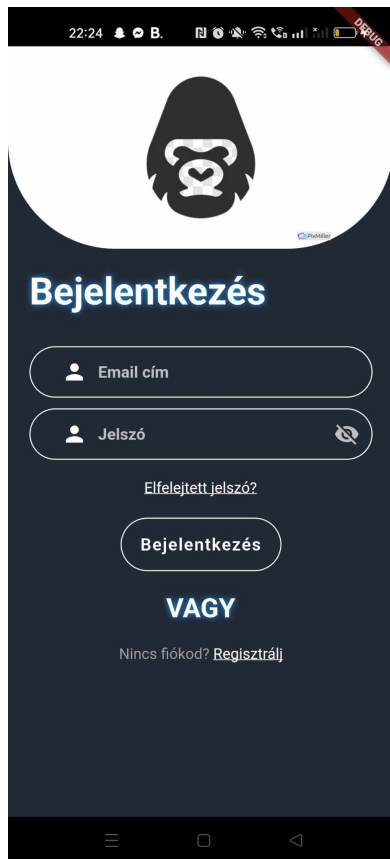
6.1.2. Regisztráció oldal

Amennyiben nem rendelkezünk regisztrált felhasználóval, a regisztrációs képernyőn megtehetjük ezt, a regisztrációs képernyőt a [6.2](#) ábrán láthatjuk, itt meg kell adnunk a következő információkat: vezetéknév, keresztnév, felhasználónév, jelszó, email cím, nem, végzettség, születési dátum. A mezők kitöltése után a regisztráció gombra kattintva készen is vagyunk, és bejelentkezhetünk a frissen létrehozott felhasználó fiókunkkal.

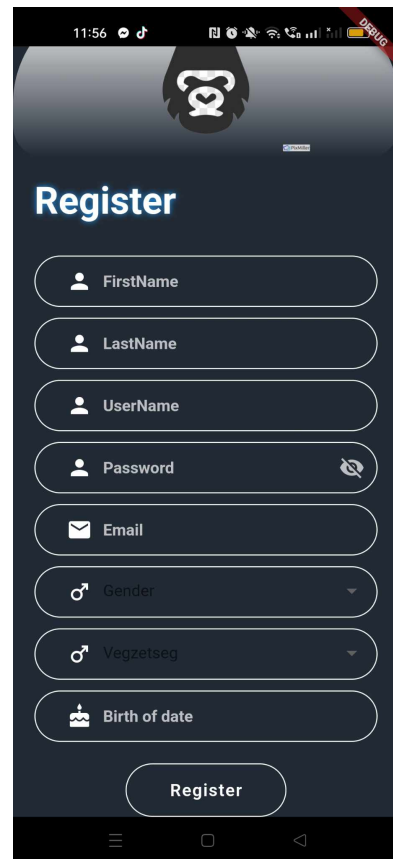
6.1.3. Menü oldal

A sikeres bejelentkezés után az alkalmazás a menübe irányít minket, innen több lehetőségünk van, ha játszani szeretnénk kiválasztjuk a megfelelő játékmódot, a játékmódok alatt található lehetőségek közül választva megváltoztathatjuk a jelszavunkat, megtekinthetjük a profilunkat, személyes adatainkat és ki is jelentkezhethetünk az alkalmazásból.

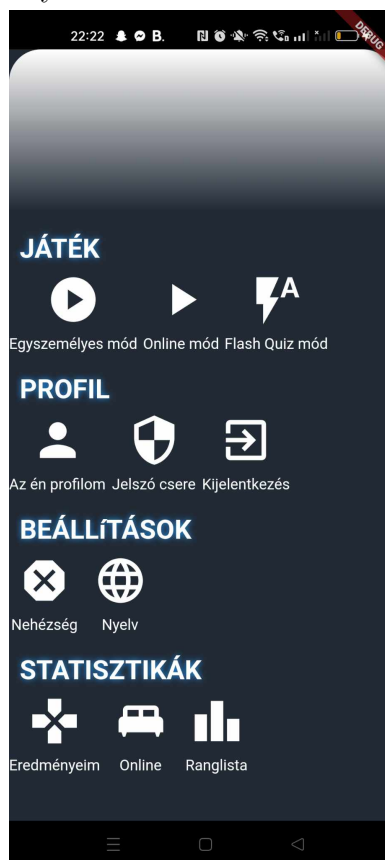
A következő ikonok segítségével kiválaszthatjuk a nehézségi szintet, ezzel állíthatjuk be hogy milyen szintű kérdéseket kapjunk, ez gyors kvíz játékmódra nem érvényes.



6.1. ábra. Bejelentkezés képernyő.



6.2. ábra. Regisztráció képernyő.



6.3. ábra. Menü képernyő.

A következő ikon egy jövőbeli fejlesztés, hogy több nyelvűvé tegyük az alkalmazást, ez még folyamatban van.

Legvégül maradtak a statisztika ikonok, itt megtekinthetjük az elvégzett tesztjeinket, eredményeinket és található még egy ranglista ikon is, amellyel megnézhetjük hogy kik a leghatékonyabb felhasználók a tesztek esetén. A menü kinézete a 6.3 ábrán látható.

6.1.4. Egyszemélyes játékmód

Az egyszemélyes játékmódra kattintva, az alkalmazás átirányít minket egy másik oldalra ahol ki kell választanunk a kívánt tantárgyat és hogy melyek fejezetből akarjuk kapni a kérdéseket, a jobb felső sarokban láthatjuk hány életünk van, ha elfogytak az életeink akkor az alkalmazás nem erre az oldalra dob be, hanem ad egy hiba üzenetet hogy nem rendelkezünk elég élettel és jöjjünk vissza később.

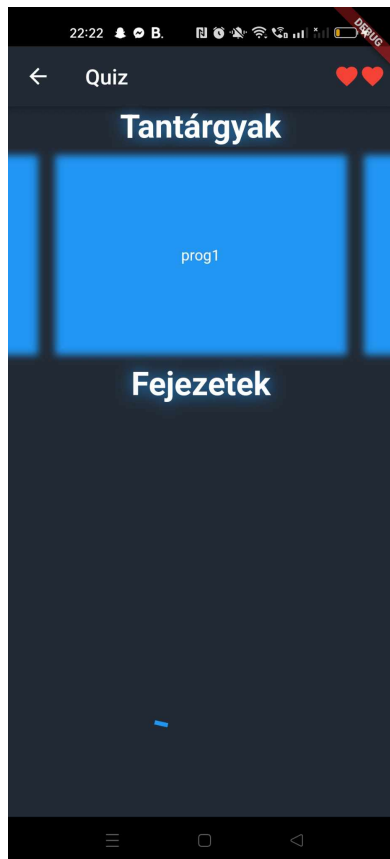
Miután kiválasztottuk a kívánt tantárgyat és fejezetet megkezdődik a játék, megkapjuk a kérdéseket, kiválasztjuk a kívánt válaszainkat és a következő gombra kattintva haladunk tovább, amennyiben helytelen válaszolunk elvesztünk egy életet, ha helyesen válaszolunk 3 kérdésre egymás után és nincs maximális számú életünk akkor kapunk egy életet. Ha elfogynak életeink a teszt közben és nem értünk a végére kapunk egy üzenetet és visszairányít minket a játék a menübe, ha pedig sikeresen elvégezzük átirányít az eredmény oldalra ahol láthatjuk az eredményünket, megmaradt életeink számát, megnézhetjük mit rontottunk el és mire válaszoltunk helyesen. Az alkalmazás ezen fázisában kerül eredményünk be a statisztika tábla, ami azt jelenti ha nem érünk a teszt végére eredményünk nem kerül be a statisztikák közé mivel sikertelen tesztként lesz tekintve. A leírt folyamatot a következő ábrák szemléltetik: 6.4 a tantárgy és fejezet kiválasztást, 6.5 az eredmény oldal kinézetét és a 6.6 ábra maga a játék menetét.

6.1.5. Gyors kvíz játékmód

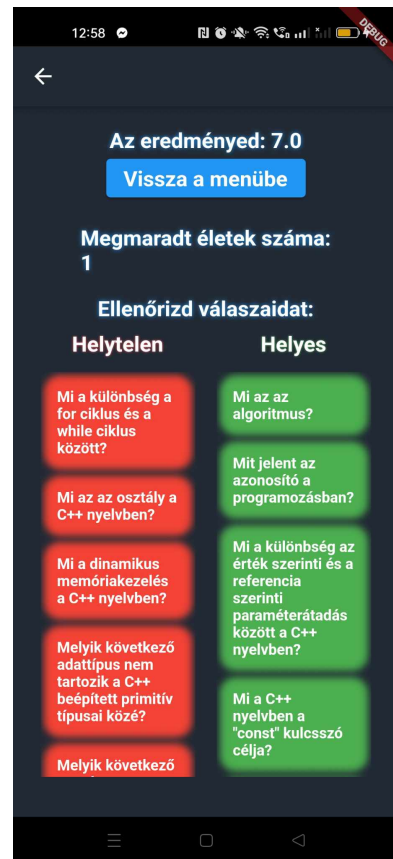
A gyors kvíz játékmód ikonra kattintva az alkalmazás átirányít minket egy oldalra ahol láthatjuk az elérhető gyors kvízeket, ha van ilyen az adott időben, ha van ilyen az indítás gombra kattintva elindul, hasonlít az előző játék menetre annyi különbséggel hogy itt nincsenek életeink. A játék végén itt is egy eredmények oldalra kerülünk, ahol megnézhetjük az elért pontszámunkat és hogy melyék kérdéseinkre mit válaszoltunk. A következő ábrák szemléltetik a játék menetét: 6.7 itt választjuk ki az elérhető gyors kvízek közül a nekünk megfelelőt, 6.9 a gyors kvíz játékmód menete és 6.8 a gyorskvíz játékban elért eredményünk.

6.1.6. Profil

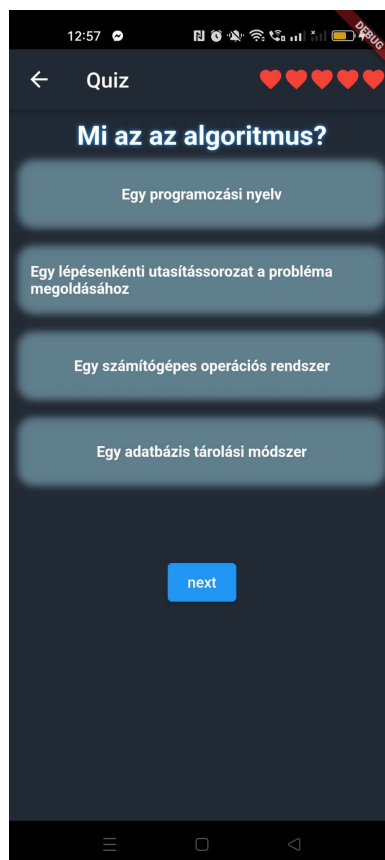
A profil fül alatt található ikonok kisebb funkcionálisokat szolgálnak mint pl. a kijelentkezés gombbal kijelentkezünk az alkalmazásból és visszadob minket a bejelentkezés oldalra. A profil ikon átirányít egy másik oldalra ahol a személyes adatainkat tekinthetjük meg és meg rendelkezik egy ikonnal ami jelszó csere, itt egy felugró ablakocskába kell beírnunk az új jelszót és következő bejelentkezéskor már azt kell használnunk.



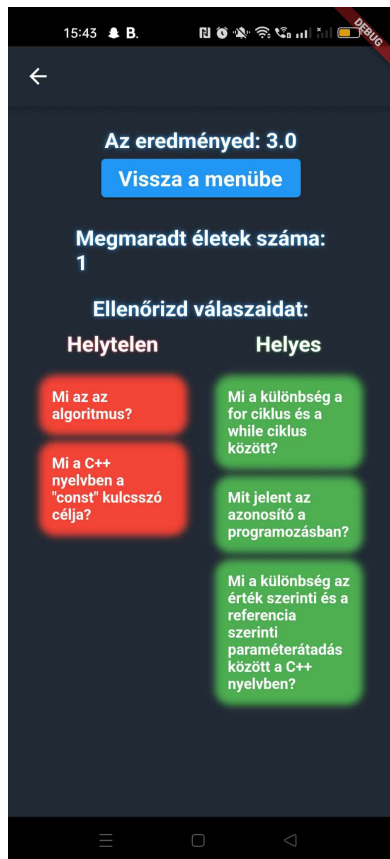
6.4. ábra. Tantárgy és fejezet kiválasztása.



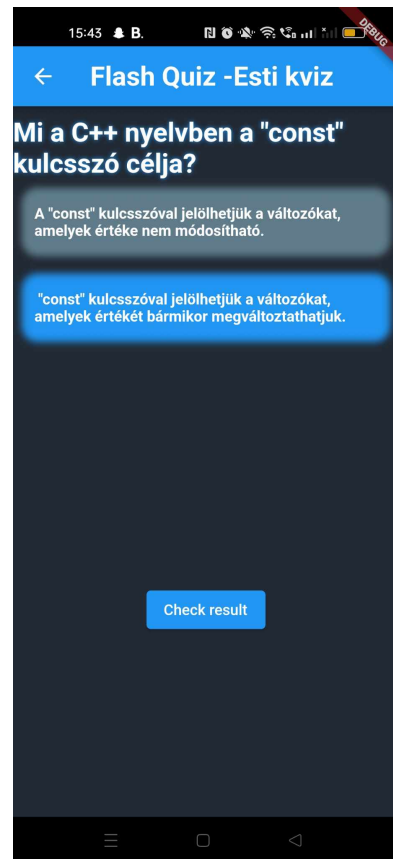
6.5. ábra. Eredmény oldal.



6.6. ábra. Játék menet oldal.



6.7. ábra. Gyors kvíz eredmény oldal.



6.8. ábra. Gyors kvíz játék menet oldal.



6.9. ábra. Gyors kvíz kiválasztása..

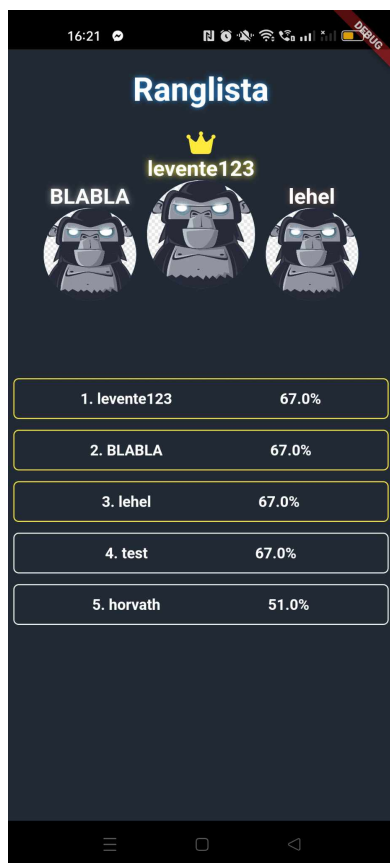
6.1.7. Beállítások

A beállítások alatt két ikont találunk, egyikkel betudjuk állítani a kérdések nehézségi szintjét, ami azt vonja maga után hogy minden tantárgyból ilyen szintű kérdéseket fogunk kapni ezt szintén egy felugró ablakocskával oldottuk meg. A másik ikon egy jövőbeli fejlesztés, a nyelveket illetően ezekről a dolgozatom során később lesz szó.

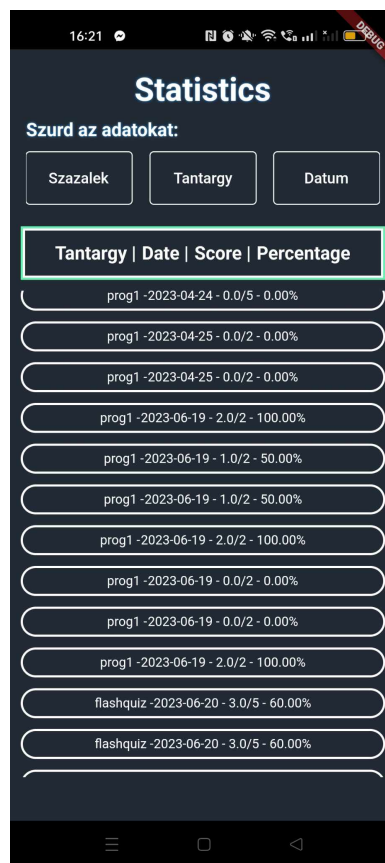
6.1.8. Statisztikák

A végső ikonok a statisztikákról szólnak, az első ikonra kattintva megtekinthetjük az általunk végzett teszteket, eredményeinket és hogy melyék teszten milyen kérdésekre mit válaszoltunk. Itt megtalálhatók a gyors kvízek statisztikái is.

A következő ikon a ranglista, ahol láthatjuk az összes felhasználót és hogy mennyire teljesítenek jól a teszteken, mégpedig az átlaguk szerint osztályozzuk őket és rendezük csökkenő sorrendbe. A harmadik ikon az online kvízek statisztikái ami szintén egy jövőbeli fejlesztés. A következő ábrák szemléltetik a statisztika oldalakat: 6.11 ábra a saját statisztikáinkról szól, míg 6.10 a ranglistát mutatja be.



6.10. ábra. Ranglista oldal



6.11. ábra. Statisztika oldal.

6.2. Adminisztrációs weboldal

6.2.1. Bejelentkezés

Az adminisztrációs weboldal is egy bejelentkezéssel indít, itt viszont nincs lehetőség a regisztrálásra, csak már adminisztrátor státusszal rendelkező személyek adhatnak hozzá új adminisztrátort, itt is egy email címet és egy jelszót kell megadnunk a bejelentkezéshez.

6.2.2. Fő oldal

A sikeres bejelentkezés után a weboldal átirányít minket a fő oldalra, ami az alapértelmezett oldal, itt néhány statisztikát látunk és a nem rég elvégzett teszteket egy táblázatban. Bal oldalt található egy menü amivel tudunk navigálni a weboldalon.

6.2.3. Adminisztrátorok oldal

Az adminisztrátorok oldalon találunk egy táblázatot ahol megvannak jelenítve az adminisztrátorok, minden adminisztrátor neve mellett található 3 gomb, lehet szerkeszteni az adataikat, lehet törölni őket és megváltoztatni a jelszavukat. Minden adminisztrátorról a következő információkat jelenítjük meg: keresztnév, vezetéknév, email cím. Ezen az oldalon található egy gomb amely átirányít minket egy új oldalra ahol létrehozhatunk új adminisztrátort megadva a fentebb említett információkat és egy jelszót.

6.2.4. Felhasználók oldal

Ez az oldal nagyon hasonló az adminisztrátorok oldalhoz, itt is ugyanaz a felépítés van egy táblázat, ahol megjelenítjük a regisztrált felhasználókat, ismét van 3 gomb, amelyek ugyanazt a célt szolgálják mint az előbb, törlés, adatok szerkesztése (itt egy felugró ablakban szerkeszthetjük az adatokat, ez igaz az egész adminisztrációs oldal szerkesztésére) és jelszó csere. Hasonlóan az oldalon van egy gomb amely átirányít minket egy új oldalra ahol létrehozhatunk új felhasználókat, megadva a következő információkat: vezetéknév, keresztnév, felhasználónév, email cím, jelszó, nem, végzettség, születési dátum.

6.2.5. Kérdések oldal

A kérdések oldal megjeleníti az összes kérdést az adatbázisból egy táblázat segítségével, keresgélhetünk köztük, szerkeszthetjük őket, törölhetjük és hozzá is adhatunk új kérdést, az új kérdés gombra kattintva amely átvissz minket egy új oldalra ahol ki kell töltenünk a kérdésre vonatkozó információkat és egy gombnyomással hozzá is adhatjuk.

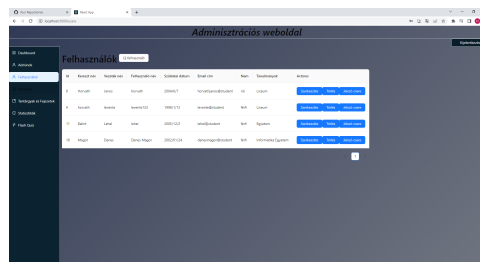
6.2.6. Tantárgyak és fejezetek oldal

A tantárgyak és fejezetek oldalon ugyancsak táblázatok segítségével kezeljük az adatbázisunkban lévő tantárgyakat és fejezeteket, a tantárgyak esetében lehet törölni, módosítani a nevüket, mivel ez az egyetlen egy adat róluk és létrehozni új tantárgyakat. Fontos megjegyezni, hogy egy tantárgy törlése maga után vonja az összes hozzá tartozó fejezet

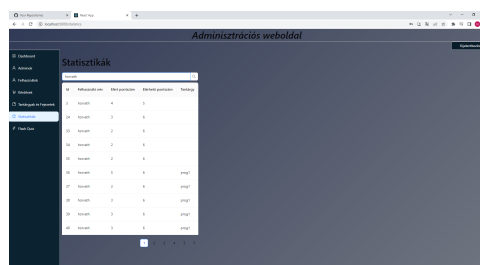
és kérdés törlését. Ugyan ezek az információk igazak a tantárgyak táblázatra is, itt a módosításnál ha egy fejezetet átrakunk egy másik tantárgyhoz akkor az összes kérdés amely az adott fejezethez tartozott átkerül az új tanárgyhoz.

6.2.7. Statisztika oldal

A statisztika oldalon megjelenítjük a felhasználók teszteken elért eredményeit, és keresgélhetünk köztük egy keresővel, a keres minden oszlop adatai között keres egyezést ami azt jelenti, hogy tantárgy, név és pontszám szerinti keresést is végezhetünk.



6.12. ábra. Adminisztrációs oldal



6.13. ábra. Adminisztrációs oldal

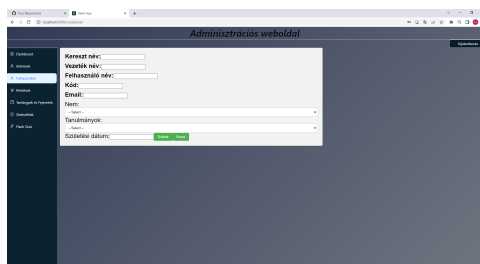
6.2.8. Gyors kvíz oldal

A gyors kvíz oldalon megjelenítjük egy táblázat segítségével az eddig létrehozott gyors kvízeket és egy gomb nyomással létre is hozhatunk egy új gyors kvízt is, hiszen a gomb nyomás átvisz minket egy másik oldalra ahol megkell adnunk a teszt nevét, a kérdéseket ki kell választanunk, hogy miket szeretnénk hozzáadni, emellett megkell adnunk két dátumot, amelyek a teszt megoldási idejét korlátozzák, vagyis csak ebben az idő intervallumban lesz megoldható a teszt.

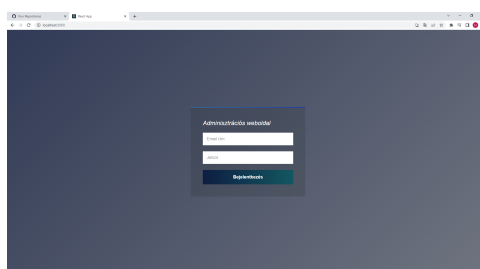
6.3. API

Az általam fejlesztett API egy egyszerű Node.js API, ez bizonyult a legnehezebb résznek a dolgozatomban, mivel a harmad év elején kevés ismeretem volt az API-okkal kapcsolatban, így ez egy igazi kihívás volt, először is bővítettem ismereteimet, olvastam hogy mit is jelent egy API, ezután technológiát kellett választani, a Node.js tűnt

a legmegfelelőbb választásnak, mivel egyetemen sikerült megismerkednem a JavaScript nyelvvel és így a Node.js szintaxisa nem volt ismeretlen számomra. Az API elkészítésében hatalmas segítség volt egy pár Youtube tanító videó. Az API modelleket használ, modellezük az adatokat s azokon hajtunk végre különböző kéréseket, ezeket a kéréseket egy vezérlő irányítja, amelyek ki exportálnak különböző műveleteket min például adatok lekérése, adatok módosítása, adat beszúrás az adatbázisba. Mindegyik ilyen vezérlő műveletet egy útvonalhoz társítjuk, amelyhez lehet címezni a kéréseket.



6.14. ábra. Adminisztrációs oldal



6.15. ábra. Adminisztrációs oldal

6.4. Tovább fejlesztési lehetőségek

A telefonos alkalmazáson és az adminisztrációs weboldalon is egyaránt rengeteget lehet javítani, így a fejlesztési lehetőségek szerintem határtalanok, főképp amit én szeretnék a jövőben mindenképp létrehozni, befejezni az több személyes játékmód, ami egy 70% körül elakadt a fejlesztésben ugyanis jelenlegi állapotban képesek vagyunk létrehozni egy szobát és két felhasználó csatlakozni tud rá de nincsenek szinkronizálva az időmérők, mivel az eredeti ötlet az volt, hogy itt majd minden kérdésre kapunk 20 másodpercet jön a következő kérdés, és folyamatosan látjuk a képernyő felső részében a saját és ellenfelünk pontszámait, de nem sikerült szinkronizálni az időmérőket és így itt elakadt a fejlesztés.

Az online lehetőség mellett nagyon jó ötletnek tartom és szeretnék az alkalmazásban több féle kérdés típust, ami azt jelenti hogy kód kiegészítő kérdéseket vagy akár kód kiértékelést alkalmazni, ami azt jelentené hogy rövid kódot írni, és az alkalmazást azt ki tudja értékelni majd így eredményt adni a felhasználó válaszára. A kellemes fejlesztői élményhez elengedhetetlen egy kellemes felhasználói felület, ezért úgy gondolom, ez is egy nagyon fontos rész lehet és talán ezzel kezdeném leghamarabb a jövőbeli fejlesztést. Ahogy

dolgozatom során említettem a jövőbeli terveim mellett az alkalmazás számára szerepel még a többnyelvűség, mivel így sokkal több diáknak válhatna hasznára az alkalmazás.

Emellett egy lokális adatbázis bevezetése, amely során a használt kérdéseket lementjük és mikor nem rendelkezünk internet kapcsolattal, ezekkel a kérdésekkel tudjunk gyakorolni, mivel jelenleg az alkalmazás csak internet kapcsolattal működik.

Végezetül pedig hogy jobban összelegyenek kötve a felhasználók egymással, egy olyan felület létrehozása az alkalmazáson belül ahol tudnak kommunikálni egymással és csoportokba, osztályokba szerveződni.

Összefoglaló

Dolgozatom során próbáltam létrehozni egy kellemes felhasználó élményt nyújtó tanító alkalmazást, amely segít jobban megérteni, bevezetést és gyakorlást biztosítani a programozás iránt érdeklődők számára.

Dolgozatomat alapos tervezéssel kezdtem el, először is a technológia kiválasztása okozta fejtörést, mivel a Flutter alkalmazásba rendelkezttem már némi tapasztalattal és kereszt platformos fejlesztést biztosít erre esett a választásom. Az adminisztrációs weboldal fejlesztéséhez React.js-re esett a választás, mivel egy gyorsan feltörekvő technológia és ki szerettem volna próbálni magam benne.

A fejlesztés folyamata nem volt zökkenőmentes, rengeteg problémával szembesültem, technikai és tervezési problémákkal egyaránt, talán a legnehezebb feladat egy teljes kép kialakítása volt az alkalmazásról. A dolgozatom bemutatását egy rövid elméleti megalkozással kezdtem, majd a követelmény specifikációkkal folytattam, ezután beszéltem a tervezésről és a felhasznált technológiákról majd bemutattam a kifejlesztett szoftvert.

Próbálkozásomat sikeresnek mondhatom, mivel sikerült létrehoznom egy olyan alkalmazást amely képes tesztek nyújtani az érdeklődők számára, felmérhetik tudásukat, nyomon tudják követni eredményeiket és részt vehetnek gyors kvízeken. Eredményeiket bármikor ellenőrizhetik, bármely megoldott tesztet visszanézhetjük, a megoldott kérdéseket és a rájuk adott válaszokat egyaránt. Az alkalmazáshoz pedig sikerült csinálnom egy adminisztrációs weboldalt amellyel folyamatosan képesek vagyunk frissíteni az adatbázis adatait így újabb és újabb tesztek tudunk előállítani. A tesztek frissítése mellett személyre szabhatjuk a felhasználókat, nyomon követni fejlődésüket és tevékenységeiket.

GitHub repository linkek:

<https://github.com/Horvath99/AllamvizsgaFlutterAlkalmazas>

<https://github.com/Horvath99/AllamvizsgaAPI>

<https://github.com/Horvath99/AllamvizsgaWebOldal>

Irodalomjegyzék

John Smith, "The Benefits of Quiz Games in Education", Educational Technology Review, 2018

Sarah Johnson, "Enhancing Interactive Learning through Quiz Applications", Journal of Educational Technology, 2021

Ábrák jegyzéke

3.1. Hasonló alkalmazások	14
4.1. Használati eset diagram (tanulók esetén)	15
4.2. Használati eset diagram (adminisztrátorok esetén)	16
5.1. A rendszer architektúrája	20
5.2. A rendszer adatbázisa	26
5.3. Telefonos alkalmazás bejelentkezés és regisztráció	29
5.4. Egyszemélyes játék szekvencia diagram	30
5.5. Gyors kvíz szekvencia diagram	31
5.6. Statisztika és profil oldal szekvencia diagram	32
6.1. Bejelentkezés képernyő.	34
6.2. Regisztráció képernyő.	34
6.3. Menü képernyő.	34
6.4. Tantárgy és fejezet kiválasztása.	36
6.5. Eredmény oldal.	36
6.6. Játék menet oldal.	36
6.7. Gyors kvíz eredmény oldal.	37
6.8. Gyors kvíz játék menet oldal.	37
6.9. Gyors kvíz kiválasztása.. . . .	37
6.10. Ranglista oldal	38
6.11. Statisztika oldal.	38
6.12. Adminisztrációs oldal	40
6.13. Adminisztrációs oldal	40
6.14. Adminisztrációs oldal	41
6.15. Adminisztrációs oldal	41