Specifikáció

# Követelmények

## Funkcionális követelmények (use-case-ek)

### OAuth2 alapú autentikáció

|  |  |  |
| --- | --- | --- |
| Azonosító | Összefoglaló | Leírás |
| F1.01 | Bejelentkezés | A felhasználó egyedi felhasználónévvel és jelszóval azonosítja magát. |
| F1.02a | Regisztráció | A felhasználó a következő adatok megadásával regisztrál: név, felhasználónév, email, jelszó. |
| F1.02b | Regisztráció adattárolás | A felhasználó által megadott adatok az adatbázisban lesznek eltárolva, a jelszó nem tiszta szövegként. |
| F1.02c | Regisztrációs szerepkörök | A felhasználó jogosultságai regisztráció után minden esetben azonosak, melyet később egy admin bővíthet. |
| F1.03a | Szerepkörök | A felhasználók különböző szerepkörökkel rendelkeznek, melyekhez eltérő jogosultságok kapcsolódnak. A szerepkörök a következők: tanuló, kutató, admin, tanár, szülő. |
| F1.03b | Szerepkör megszemélyesítése | A tanári és szülői szerepkörrel rendelkező felhasználók megszemélyesíthetnek egy tanuló felhasználót, és annak nevében járhatnak el. |
| F1.04 | Külső alkalmazások számára autentikáció | Az autentikációs szerver bizonyos előre definiált külső alkalmazásokkal is kapcsolatban áll, azok kérésére elvégzi a felhasználók azonosítását. És rendelkezésre bocsátja a felhasználó adatait. |
| F1.05 |  |  |

### A játékosok képességeinek tárolása

|  |  |  |
| --- | --- | --- |
| Azonosító | Összefoglaló | Leírás |
| F2.01 | Játékos kognitív profiljának tárolása | Minden játékoshoz az összes CHC képesség és ahhoz tartozó legfrissebb érték lesz eltárolva. |
| F2.02 | Egyéb képesség tárolása | A játékosokhoz más tetszőleges értéket felvevő (pl. bináris) képesség is tárolható. |
| F2.03 | Képességek visszakövethetősége | Az alkalmazás lehetőséget biztosít a játékos tetszőleges időpontbeli képességének lekérdezésére. |
| F2.04 | Új képesség felvétele | A játékoshoz új képességeket is fel lehet venni |

### Játékok kezelése

|  |  |  |
| --- | --- | --- |
| Azonosító | Összefoglaló | Leírás |
| F4.01 | Játékok adatainak tárolása | Az alkalmazás külső játékokhoz tartozó információkat tárol. Az ehhez tartozó adatok: azonosító, érintett képességek, nehézség, elérési útvonal (url) |
| F4.02 | Játékok adatainak frissítése | Az játékosok eredményei alapján a játékok nehézsége folyamatosan frissül. A játékok egymásra is hatnak (ez adattárolás szintjén jelentkezik-e bárhogy)? |
| F4.03 | Játékok elérése | Az applikációból egy hivatkozásra klikkelve a játék egy új oldalon elindul. |
| F4.04 | Játékok számára biztosított adatok | A játékost közvetlenül azonosító adat nem hozzáférhető a játék számára (név, email, felhasználónév)  A játék lekérdezheti a játékos képességeit, de csak az aktuális játék az aktuális játékos képességeit. (Ehhez az alkalmazás hozzáférési tokent generál a játék számára, mellyel az azonosítja magát.) |
| F4.05 | Játékmenetek elkülönítése | A játékokat játékmenet azonosító különbözteti meg, melyet új játék esetén a játék generál. |
| F4.06 | Játék által küldött adatok | A játék minden játékmenethez adatokat küldhet az alkalmazás számára. Az adatküldéshez szintén azonosítja magát. A küldött adat tartalmazza a játékmenet azonosítót és a játék eredményét (bármilyen mért érték). |
| F4.07 | Játékmenetek és játékosok összekapcsolása | Minden felhasználóhoz az összes általa játszott játékmenet adatai el lesznek tárolva. A játékmenetekről az alábbi adatok: játék verzió, időpont, eredmény. |
| F4.08 | Játék keresése | Jogosult felhasználók különböző szempontok alapján kereshetnek a rendelkezésre álló játékok között. |
| F4.09 | Adatmentés játékindításkor | Egy játék elindításakor a játékos aktuális kognitív adatai el lesznek mentve időbélyeggel ellátott rekordként. Legfeljebb napi egyszer készül ilyen mentés, amikor a profilozós játéklehetőséget használja a felhasználó. |
| F4.10 |  |  |

### Játékok ajánlása

|  |  |  |
| --- | --- | --- |
| Azonosító | Összefoglaló | Leírás |
| F5.01 | Játék elérésének lehetőségei | A játékos többféleképp érhet el egy játékot: egy arra jogosult felhasználó beállít számára egy konkrét játékot vagy a rendszer ajánl számára egy játékot |
| F5.02 | Játék kijelölése egy másik felhasználónak | Az arra jogosultak (pl. tanár, kutató) kijelölhetnek egy vagy több játékot más felhasználók számára. |
| F5.03 | Játék ajánlása képesség alapján | Az alkalmazás a játékos képességének megfelelő nehézségű játékot ajánl. |
| F5.04 | Profil és tréning elkülönítése | Amennyiben a játékosnak hiányos a profilja, azaz nem minden képességéhez tartozik érték, a rendszer profil teszteket ajánl, melyek a képességeket mérik.  Teljes profil esetén tréning teszteket ajánl  Tréning: nem számít bele a képességszámításba, növekvő nehézségűek  Profil: beleszámít a képességszámításba, a felhasználó kiválaszthatja a nehézséget  Tréninget naponta akármennyiszer próbálhat a felhasználó, profilfejlesztő játékot viszont csak egyszer. |

### Frontend

|  |  |  |
| --- | --- | --- |
| Azonosító | Összefoglaló | Leírás |
| F6.01 | Nyitóoldal | Legyen regisztráció, bejelentkezés gomb figyelemfelkeltő háttérrel. A gombokra kattintva átirányít az OAuth2-es azonosításra |
| F6.02 | Regisztráció | A felhasználó saját adatokkal (név, email, felhasználónév, jelszó) tud regisztrálni, ezeket egy adatbázisban menti el az alkalmazás |
| F6.03 | Bejelentkezés | A felhasználó a korábban regisztrált felhasználónév/jelszó megadásával be tud jelentkezni az alkalmazásba. |
| F6.04 | Profil | A felhasználó meg tudja nézni az aktuális/korábbi napokon elmentett képességeit a korábbi játékok alapján a jobb felső sarokban található ikonra kattintva. Lehetne grafikon, ami mutatja a képességek alakulását az idő függvényében |
| F6.05 | Játékok listája | A felhasználó tud böngészni a játékok között, 3 menüpont:   * minden játék (lehessen random ajánlást kérni) * tanárok által javasolt játékok (legördülő menü, több tanár is javasolhat játékot, mindegyikhez ki tudjuk választani, hogy az általa javasoltakat mutassa) * kutatók által javasolt játékok (legördülő menü, több kutató is javasolhat játékot, mindegyikhez ki tudjuk választani, hogy az általa javasoltakat mutassa)   Mindegyik oldalon tudja elindítani a játékokat, valamint mutassa a legutóbbi és a legjobb eredményét az adott játékban  Lehessen szűrni képesség alapján |
| F6.06 | Admin felhasználói adminisztráció | Az összes felhasználót látja egy táblázatban, itt tudja módosítani bárkinek a szerepkörét:   * hozzá tud adni/el tud venni tanári/kutatói/szülői jogosultságokat * hozzá tud rendelni/el tud venni megszemélyesíthető felhasználót (legördülő lista kereső mezővel, checkboxot bepipálja) * jelszó felülírása (extra megerősítést kér felugró ablakban) |
| F6.07 | Admin játéklista adminisztráció | Az összes játék adatait látja egy táblázatban és tudja módosítani őket:   * Ikon (thumbnail) * Név * Leírás * Verzió * Aktivitás (ezt be tudja állítani, hogy lehet-e vele játszani, ekkor aktív, vagy sem) * Link * JSON leíró (mivel lehet felparaméterezni, mik a szélsőértékek (pl. nehézségi szintnél), melyik paraméter melyik kognitív skála értékből hogy számolódjon (azt is figyelembe véve, hogy a skála is adaptív és változhat), az eredmény értékeknek (pontszámoknak) mik a szélsőértékeik, melyik eredmény érték melyik kognitív képességre hat) |

### Játékok megvalósítása

|  |  |  |
| --- | --- | --- |
| F7.01 | Technológia | A játékok megvalósítása a Phaser.js keretrendszer segítségével történik. |
| F7.02 | Képességek API | A játékok képesek kommunikálni az alkalmazás backendjével, hogy lekérdezhessék vagy módosíthassák a játékosok CHC képességértékeit. |
| F7.03 | OAuth2 Token | A játékok a hozzáférési tokent használják az alkalmazás backendjéhez való hitelesítéshez. Ezt a tokent az alkalmazás generálja, és csak az aktuális játékos és az aktuális játék számára lesz elérhető. |
| F7.04 | Játékmenet Azonosító | Minden egyes játékmenet generál egy egyedi azonosítót, amely segít a játékmenetek és a játékosok összekapcsolásában az adatbázisban. |
| F7.05 | Adat Küldése | A játékok képesek adatokat küldeni az alkalmazás backendjéhez a játékmenet és a játékos teljesítményével kapcsolatban. Ezen adatok tartalmazhatják a játékmenet azonosítót és a játék eredményét. |
| F7.06 | Dinamikus Nehézség | A játékok alkalmazkodnak a játékosok képességszintjéhez, és dinamikus nehézséget állítanak be. |
| F7.07 | Játék Frissítések | A játékok támogatják az adatbázisban tárolt játék adatainak frissítését, például a nehézségi szint vagy érintett képességek változásait. |
| F7.08 | Adatvédelmi Protokoll | A játékok soha nem férnek hozzá a játékosok közvetlenül azonosító adataihoz (név, email, felhasználónév). |
| F7.09 | Képességek Frissítése | A játékok befejezése után automatikusan frissülnek a játékosok képességadatai az alkalmazás adatbázisában. |
| F7.10 | Hiba Kezelés | A játékok tartalmaznak hibakezelési logikát, hogy az esetleges hibák ne befolyásolják az alkalmazás egyéb funkcióit. |

## Műszaki követelmények

# Részletes terv

## Vizuális tervek (elérhető oldalak) – ezt megcsinálom Paintben (Matyi)

## Adatmodell - ezt megcsinálom (Anna)

### Entitások

Az alkalmazásban adattárolás alapjául szolgáló entitások a következők:

* Felhasználó: a rendszerben megjelenő aktor, a felhasználók között ismerettségi viszony áll fenn (ez teszi lehetővé a megszemélyesítést, a játékok ajánlását)
* Szerepkör: a felhasználók jogosultságait összefoglaló szerep
* Játékmenet: a felhasználó által adott időpillanatban játszott egyetlen játék eredménye
* Játék: profilozó vagy tréning játék, mely kognitív képességeket mér/fejleszt, több nehézségi szintje lehet
* Ajánlott játék: azon játékok listája, melyet egy arra jogosult felhasználó egy másik felhasználónak ajánl/előír
* Képességprofil: a felhasználó aktuális profil-képességeit írja le
* Képesség-snapshot: a felhasználó adott időpillanatban értelmezett képesség-profilja

### Adatbázis adatkapcsolat diagramja

A képen szöveg, diagram, Tervrajz, Műszaki rajz látható

Automatikusan generált leírás

## Interfészek (REST API) - ezt megcsinálom (Matyi)

## OAuth2 – ezt szívesen megcsinálom (Anna)

Az elkészült szoftver OAuth2 autentikációt és autorizációt biztosít, mely lehetővé teszi, külső alkalmazások számára a felhasználók azonosítását, és jogosultságaik elérését. Az OAuth 2.0 specifikációban ismertetett szerepeknek megfelelően dedikált autorizációs szerver (a továbbiakban Authorization Server) lesz megvalósítva, illetve egy erőforrás szerver (a továbbiakban Resource Server) és egy kliens alkalmazás. Az első kettő Spring Security 6 keretrendszer felhasználásával készül el, míg a kliens (…?)

### Authorization Server

A komponens egy külön backendként funkcionál, melynek feladata a felhasználók azonosítása és jogosultságaik kezelése, valamint a kliens alkalmazások számára autorizáció biztosítása. Ez az alkalmazás a terhelés elosztása miatt különválik a projekt során elkészített többi komponenstől, önállóan futtatható.

Az alkalmazáshoz külön adatbázis kapcsolódik, melyben a felhasználók azonosításhoz szükséges adatai lesznek eltárolva (név, id, secret, jogosultságok). Emellett a regisztrált kliensalkalmazások azonosítására alkalmas adatai is innen elérhetők. A jelszóként funkcionáló titkos adatok az adatbázisba hashelt formában íródnak. Az adatbázis relációs sémára épül.

#### Regisztráció

A szerver regisztrációs felületet biztosít új felhasználók számára. A felhasználói adatok megadását követően új entitás kerül az adatbázisba. A felhasználó jogosultságait később az erre jogosult felhasználók állíthatják be.

#### Bejelentkezés

A felhasználót egy általa használt egy kliensalkalmazás átirányítja a bejelentkezési felületre, ahol a felhasználó az egyedi felhasználónevét és jelszavát megadva azonosítja magát. Sikeres bejelentkezést követően a szerver a kliens által megjelölt url-re irányít és elküldi a hozzáférési tokent.

A token egy jwt token, melybe a felhasználó neve, azonosítója és jogosultságai vannak kódolva. A kliens ezt a tokent használhatja a Resource Serverhez intézett kéréseknél azonosításra.

#### Kliensek azonosítása

Az Authorization Servert csak előre meghatározott kliensalkalmazások érhetik el, ezeket előre regisztrálni kell. A regisztrált kliensek azonosítót és titkos adatot kapnak, mellyel azonosíthatják magukat. Emellett a klienshez tartozó átirányítási url-ek is előre definiáltak.

### Resource Server

Ez egy REST API szerver, mely bizonyos privát endpoint-jait védi. Ezekhez csak azonosított és megfelelő jogosultsággal rendelkező felhasználók férhetnek hozzá.

Az alkalmazás a kérések fejlécében jwt-tokent vár, melyet az Authentication Server segítségével validál. A Resource Server nem kezeli a felhasználók jogosultságait, csupán felhasználja az Authentication Server-től kapott információt. Érvényes token esetén kiszolgálja a kérést.

### Client

A kliens alkalmazás az Authentication Server-t felhasználva igényel hozzáférési tokent a felhasználó számára. Ehhez egy kliens ID-val, titkos adattal és átirányítási címmel (ide várja a válasz tokent) felparaméterezett kérést intéz az Authentication Server felé. A válaszban kapott jwt-tokent felhasználja a Resource Server felé intézett kéréseknél.

### Szerepkörök, jogosultságok

Minden felhasználó egy vagy több szerepkörrel rendelkezik, melyek különböző jogosultságokat biztosítanak. Az Authentication Server kizárólag a felhasználók szerepkörét tárolja, és azonosítás után ezt továbbítja a Resource Server (illetve más külső alkalmazás) számára. A szerepkörökhöz tartozó jogosultságokat a Resource Server oldja fel, mivel ez minden alkalmazásnál eltérő lehet, és idővel változhat.

Bizonyos felhasználói szerepkörök (pl. tanár, szülő) más felhasználókat személyesíthetnek meg (pl. tanuló). A megszemélyesítés az Authentication Server szintjén történik. Miután a felhasználó azonosította magát, a sikeres belépés után az alkalmazás felkínálja (amennyiben lehetséges), hogy mely felhasználó nevében szeretne bejelentkezni. Emiatt a kiváltságos felhasználókhoz további felhasználók vannak rendelve az adatbázisban, akiknek a szerepében felléphet. Megszemélyesítés esetén az Authentication Server olyan hozzáférési tokent továbbít a Resource Server számára, melyben a megszemélyesített felhasználó adatai, illetve egy megszemélyesítést jelző flag szerepel.

Az adatbázisban tárolt szerepkör lehetséges értékei a következők: tanuló, kutató, admin, tanár, szülő.

## Játékok megvalósítása

Az egyszerű 2D képességfelmérő játékok a Phaser.js keretrendszer segítségével kerülnek kifejlesztésre. A játékok célja, hogy mérjék és értékeljék a felhasználók különböző képességeit, például memóriát, koordinációt vagy logikai gondolkodást. A játéklogika és a grafikai megjelenítés teljes mértékben a Phaser.js-en alapszik, amelynek előnye az egyszerűség és a gyorsaság.

A felhasználói adatok, mint például a pontszámok és az elért szintek, egy Spring backenddel lesznek szinkronizálva. A funkcionális követelményeknek megfelelően, a játékok többféle nehézségi szinttel és változatos feladatokkal rendelkeznek. A játékállapotok és a felhasználói eredmények RESTful API-k segítségével kerülnek átküldésre a frontend és a backend között.

A backend az adatokat egy adatbázisban tárolja, amely lehetővé teszi a felhasználók teljesítményének hosszú távú nyomon követését. Az adatok JSON formátumban kerülnek átküldésre, hogy egységes és gyors adatkezelés legyen biztosítva.

## Technológiák, keretrendszerek

* Kotlin
* SpringBoot 3.1.3
* Angular
  + Bootstrap - Keressünk
* MySQL
* Spring Security
* Gradle 8.2
* Phaser.js 3.60.0