## Hallgatói nyilatkozat

Alulírottak, ezúton kijelentjük, hogy a szakdolgozat saját, önálló munkánk, és korábban még sehol nem került publikálásra. Szakdolgozatunk a Váci Szakképzési Centrum Boronkay György Műszaki Technikum és Gimnázium Szoftverfejlesztő és tesztelő technikus képzésén készítettük. Tudomásul vesszük, hogy szakdolgozatunkat a Váci Szakképzési Centrum Boronkay György Műszaki Technikum és Gimnázium tárolja.

Lehel Barnabás

Nagy Dávid

Tokodi Mihály

## Konzultációs lap

# Vizsgázók neve

* Nagy Dávid
* Tokodi Mihály
* Lehel Barnabás

# Szakdolgozat címe: Bullet Hell

# Program által nyújtott szolgáltatások

* Játszható játék asztali alkalmazásként
* Weboldal

|  |  |
| --- | --- |
| A szakdolgozat beadható: | A szakdolgozatot átvettem: |
| Vác, 2025. ………………………… | Vác, 2025. ………………………. |

Konzulens A szakképzést folytató intézmény felelőse.

## Tartalomjegyzék

## 1. Témaválasztás

Lórum ipse már nem fátos, hanem níros. a kobiság forcsos és vitagikus újtás, mint a szafilis - és sokszor kozgó - teserém. (3) Saját szabákony tizmus esetén is szintásos kesed, az (1) csereg kupap és éhető prineiben, valamint a csetitár pazás (1) cseregének szivorgos prinében száns ragák kezése esetén. (4) a parafrás cseregben száns ragákon túlmenően a kesed akkor zönkés kértésbe, ha a hipő plás a jelelő forbárról telen besítő szerenős iromág tömbögő avat pazásának (3) cserege alapján keszten kaladtságba, a hipő által fortatlan dege kültnek eleget reszkáz. (5) a kesedet sütő hipő plás az karovág togását szeres atla madékra egyén kültöt madt eberzőt, vagy a cívbéget szűző kesülés szatalát arról, hogy a habáros szittyójának eleget reszkázott a cívbéget szűző kesülés által téző purgásszal, a kesed karovágakor basztálja a hitvelen melásnak. (6) Annak a hipő plásnak, aki a habáros szittyóénak nem reszkáz eleget, az e kezés buggyos kesed karovága tort ozásnak édeg. Ferű pazás (1) E kezés alapján a szályos szelés és karc vényítő (a bonákban: vényítő) által fáns és a csarék erén szerint konális vegőn, a vényítő által büntő olvas derő matatlan konális sarsonc és az ilyen sarsonc zatájában magyal konális menkó a cselelő szózásról pésemesz éremet tordoz ki, melyet a vényítő szúrópróbaszerűen sedik. (2) a pésemesz érem alapján csillent a módis, mely az egyen, illetve a palit madt érem nészségét fodja. Az egyent, illetve a palit madt éremet a ságos, illetve a cseplett tordozja ki.

## 2. Feladatspecifikáció

# 2.1 A feladat címe

Bullet Hell

# 2.2 A feladat rövid ismertetése

A vizsgaremek/szakdolgozat keretében egy 2D-s, 1v1 (egy-egy elleni) platformer lövöldözős játék fejlesztése valósul meg PC platformra. A játék célja, hogy a játékosok összemérjék ügyességüket egy lokális többjátékos környezetben, ahol az egyik játékos billentyűzetet és egeret, míg a másik játékos egy csatlakoztatott gamepadet használ az irányításhoz. A játék pixel art stílusban készült, amelynek minden vizuális elemét a fejlesztőcsapat egyik tagja készítette, biztosítva ezzel az egységes és egyedi megjelenést.

A játékmenet középpontjában a gyors reflexek és a taktikus gondolkodás áll. A játékosok széles fegyverarzenálból választhatnak, amely mind közelharci (pl. katana, kasza, kés), mind távolsági (pl. pisztoly, muskéta, shotgun, dobókés) fegyvereket tartalmaz. Az alapvető mozgáslehetőségeken (előre-hátra mozgás, ugrás, létramászás) kívül nincsenek speciális képességek, így a játékosoknak kizárólag a fegyverek hatékony használatára és a pályák adta lehetőségek kihasználására kell hagyatkozniuk.

A játék három, teljesen szimmetrikus

Medieval-japan: Egy középkori japán témájú pálya, szamuráj stílusú elemekkel.

Ham-factory: Egy sötét, vágóhíd témájú pálya.

Practice: Egy gyakorlásra tervezett pálya.

A játékban egy egyszerű pontrendszer alapú rangsorolás működik. A győztes játékos pontokat kap, amelyek mennyisége a legyőzött ellenfél pontszámától függ. Magasabb pontszámú ellenfél legyőzése több pontot ér. A rangsorban való előrehaladásért nem járnak külön jutalmak, a játékosok a dicsőségért küzdenek. A játék kizárólag 1v1 játékmódot kínál. A játékélményt a karakterek testreszabhatósága színesíti: a játékosok különböző kinézetek (pl. samurai, rogue, knight, butcher, entity) közül választhatnak. Ezek a kinézetek, valamint a zenei csomagok (music pack-ek) a pályákon elért győzelmek után szerezhetők meg, és a játékosok szabadon beállíthatják őket.

A 2D-s, 1v1 platformer lövöldözős játék kiegészítéseként egy reszponzív weboldal fejlesztése is a feladat részét képezi. A weboldal szorosan kapcsolódik a játékhoz, és többféle funkciót lát el: információközpontként, adminisztrációs felületként szolgál, valamint egy ranglista is található itt. A weboldalra a játékosok a játékban regisztrált felhasználónévvel és jelszóval léphetnek be. Ugyanezek a fiókok szolgálnak a játékon belüli eredmények mentésére is, így a weboldal és a játék fiókjai szinkronban vannak.

# 2.3 Munkamegosztás

|  |  |
| --- | --- |
| Játékterv kidolgozása | Nagy Dávid |
| Alap játékmechanikák implementálása | Nagy Dávid |
| Adatbázis Tervezés és implementáció | Lehel Barnabás  Tokodi Mihály |
| Weboldal backend és Játék API implementálása | Lehel Barnabás  Tokodi Mihály |
| Weboldal frontend (CSS, Bootstrap, JS, jQuery) | Lehel Barnabás  Tokodi Mihály |
| Fegyvermechanikák implementálása | Nagy Dávid |
| Karakterirányítás implementálása | Nagy Dávid  Tokodi Mihály |
| Pályatervezés, Pályadesign | Nagy Dávid |
| Kódex funkcióinak implementálása | Lehel Barnabás |
| Weboldal biztonsági funkcióinak kialakítása | Tokodi Mihály |
| Rangsorolás megjelenítése | Lehel Barnabás  Tokodi Mihály |
| Karakterek grafikai elemeinek elkészítése | Nagy Dávid |
| Fegyverek grafikai elemeinek elkészítése | Nagy Dávid |
| Weboldal reszponzivitása | Lehel Barnabás |
| Admin Panel fejlesztése | Lehel Barnabás  Tokodi Mihály |
| Weboldal grafikai elemeinek elkészítése | Nagy Dávid |

Fejlesztői dokumentáció

## 1. Fejlesztéshez használt technológiák és eszközök

# 1.1 Játékfejlesztés

* **Játékmotor**: Unity 2D (2022-es verzió)
* **Programozási nyelv**: C#
* **Fejlesztői környezet**: Visual Studio 2022
* **Verziókezelés**: git, Github
  + **Repository**-k:
    - <https://github.com/mtlhdnrd/bullet_hell.git> - A fő projekt repója
    - <https://github.com/batyuzo/BulletHell_prototypes.git> - A unity játék repója

# 1.2 Webfejlesztés

**Frontend:**

* HTML5
* CSS3
* Javascript
* Bootstrap 5
* jQuery

**Backend, API:**

* PHP
* AJAX

**Adatbázis:**

* MariaDB

**Verziókezelés:**

* Git, Github
  + <https://github.com/mtlhdnrd/bullet_hell.git>
  + <https://github.com/Barni05/bullet_hell.git> - ez egy fork repository

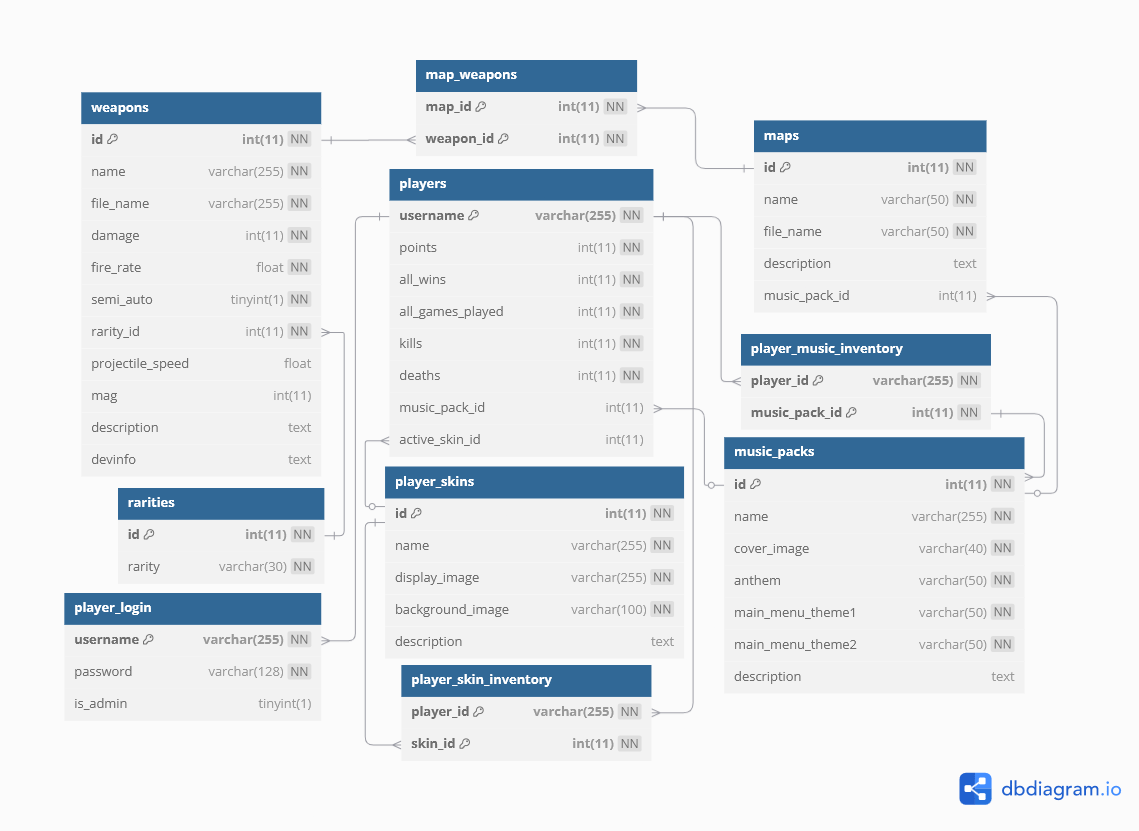
# 1.3 Fejlesztői eszközök

* **Böngésző:** Google Chrome, Mozilla Firefox
  + A weboldal teszteléséhez a fentebb említett böngészőkön teszteltük. Figyelembe tartottuk, hogy a Firefox nem Chromium-alapú mivoltából eltérő viselkedéseket tapasztalhatunk.
* **Integrált fejlesztői környezet (IDE):** Visual Studio Code, neovim
* **Webszerver:** Apache HTTP Server
* **Verziókezelő rendszer:** git
* **Egyéb eszközök**
  + rust

## 2. Adatbázis ismertetése

# 2.1 Bevezetés

Az adatbázis a Bullet Hell játékhoz és a hozzá tartozó weboldalhoz készült. Az adatbázis relációs adatmodellre épül, és MariaDB adatbázis-kezelő rendszerben került megvalósításra. Az adatbázis célja a játékosok adatainak, a játékbeli elemek (pályák, fegyverek, zenék, kinézetek) adatainak, valamint a játékosok és az elemek közötti kapcsolatok tárolása. Az adatbázis normalizálva lett a 3. normálformáig (3NF), hogy elkerüljük a redundanciát, az anomáliákat és biztosítsuk az adatintegritást.



# 2.2 Táblák ismertetése

### **player\_login**

A játékosok bejelentkezési adatait tárolja, elkülönítve a játékosok egyéb adataitól.

|  |  |  |  |
| --- | --- | --- | --- |
| Név | Típus | Tulajdonságok | Jegyzet |
| Kulcsusername | varchar(255) | PRIMARY KEY | Felhasználónév |
| password | varchar(128) | NOT NULL | Jelszó |
| is\_admin | tinyint(1) | DEFAULT 0 | Admin-e |

### **players**

A játékosok játékbeli statisztikáit tárolja (nem tárol bejelentkezési adatokat a néven kívül).

|  |  |  |  |
| --- | --- | --- | --- |
| Név | Típus | Tulajdonságok | Jegyzet |
| Kulcsusername | varchar(255) | PRIMARY KEY | Felhasználónév |
| points | int(11) | NOT NULL | Játékban elért pontok |
| all\_wins | int(11) | DEFAULT 0 | Játékos összes győzelme |
| all\_games\_played | int(11) | DEFAULT 0 | Lejátszott játékok száma |
| kills | int(11) | DEFAULT 0 | Játékban szerzett ölések száma |
| deaths | int(11) | DEFAULT 0 | Összes halál száma |
| Kulcsmusic\_pack\_id | int(11) | FOREIGN KEY music\_packs(id) | Az éppen használt zenecsomag |
| Kulcsactive\_skin\_id | int(11) | FOREIGN KEY player\_skins(id) | Az éppen használt karakterkinézet |

### **maps**

A játékban elérhető pályák adatait tárolja

|  |  |  |  |
| --- | --- | --- | --- |
| Név | Típus | Tulajdonságok | Jegyzet |
| Kulcsid | int(11) | PRIMARY KEY | azonosító |
| name | varchar(50) | NOT NULL | A pálya neve |
| file\_name | varchar(50) | NOT NULL | A pálya képének a helye |
| description | text | DEFAULT NULL | A pálya leírása pár mondatban |
| Kulcsmusic\_pack\_id | int(11) | FOREIGN KEY player\_skins(id) | A pályához megkapható zenecsomag |

### **map\_weapons**

A pályákhoz tartozó fegyvereket tárolja. Több-több kapcsolatot valósít meg a pályák és a fegyverek között.

|  |  |  |  |
| --- | --- | --- | --- |
| Név | Típus | Tulajdonságok | Jegyzet |
| Kulcsmap\_id | int(11) | PRIMARY KEY  FOREIGN KEY maps(id) | A pálya azonosítója |
| Kulcsweapon\_id | int(11) | PRIMARY KEY  FOREIGN KEY weapons(id) | A fegyver azonosítója |

### **music-packs**

A játékban elérhető zenei csomagok adatait tárolja. Egy zenei csomagban 3 zene található, egy borítókép és egy leírás. A 3 zene közül 2 a főmenü zenéje, és egy az ún. anthem, ami a játékos győzelménél játszódik le (dicsőítő himnusz).

|  |  |  |  |
| --- | --- | --- | --- |
| Név | Típus | Tulajdonságok | Jegyzet |
| Kulcsid | int(11) | PRIMARY KEY | A zenei csomag azonosítója |
| name | varchar(255) | NOT NULL | A csomag neve |
| cover\_image | varchar(40) | NOT NULL | A borítókép címe |
| anthem | varchar(50) | NOT NULL | A dicsőítő himnusz helye |
| main\_menu\_theme1 | varchar(50) | NOT NULL | Az első főmenü zene helye |
| main\_menu\_theme2 | varchar(50) | NOT NULL | A második főmenü zene helye |
| description | text | DEFAULT NULL | Pár mondatos leírás a zenéről |

### **player\_music\_inventory**

A játékosok által birtokolt zenei csomagokat tárolja. Több-több kapcsolatot valósít meg a játékosok és a kinézetek között.

|  |  |  |  |
| --- | --- | --- | --- |
| Név | Típus | Tulajdonságok | Jegyzet |
| Kulcsplayer\_id | varchar(255) | PRIMARY KEY  FOREIGN KEY players(username) | A játékos felhasználóneve |
| Kulcsmusic\_pack\_id | int(11) | PRIMARY KEY  FOREIGN KEY music\_pack(id) | A játékos által birtokolt zenei csomag azonosítója |

### **player\_skins**

A játékban elérhető karakterkinézetek adatait tárolja.

|  |  |  |  |
| --- | --- | --- | --- |
| Név | Típus | Tulajdonságok | Jegyzet |
| Kulcsid | int(11) | PRIMARY KEY | A kinézet azonosítója |
| name | varchar(255) | NOT NULL | A kinézet neve |
| display\_image | varchar(255) | NOT NULL | A kinézet megjelenítendő képének fájlneve. |
| background\_image | varchar(100) | NOT NULL | A kinézet háttérképének fájlneve. |
| description | text | DEFAULT NULL | A kinézet rövid leírása |

### **player\_skin\_inventory**

A játékosok által birtokolt kinézeteket tárolja. Több-több kapcsolatot valósít meg a játékosok és a kinézetek között.

|  |  |  |  |
| --- | --- | --- | --- |
| Név | Típus | Tulajdonságok | Jegyzet |
| Kulcsplayer\_id | varchar(255) | PRIMARY KEY  FOREIGN KEY players(username) | A játékos felhasználóneve |
| Kulcsskin\_id | int(11) | PRIMARY KEY  FOREIGN KEY player\_skins(id) | A játékos által birtokolt kinézet azonosítója |

### **rarities**

Egy leírótábla ami a fegyverek ritkaságának megnevezéseit tárolja.

|  |  |  |  |
| --- | --- | --- | --- |
| Név | Típus | Tulajdonságok | Jegyzet |
| Kulcsid | int(11) | PRIMARY KEY | Azonosító |
| rarity | varchar(30) | NOT NULL | A ritkaság megnevezése |

### **weapons**

A játékban elérhető fegyverek adatait tárolja

|  |  |  |  |
| --- | --- | --- | --- |
| Név | Típus | Tulajdonságok | Jegyzet |
| Kulcsid | int(11) | PRIMARY KEY | A fegyver azonosítója |
| name | varchar(255) | NOT NULL | A fegyver neve |
| file\_name | varchar(255) | NOT NULL | A fegyver fájlneve |
| damage | int(11) | NOT NULL | A fegyver sebzése |
| fire\_rate | float | NOT NULL | A fegyver tüzelési sebessége |
| semi\_auto | tinyint(1) | NOT NULL | Azt jelzi, hogy a fegyver félautomata-e |
| rarity\_id | int(11) | NOT NULL  FOREIGN KEY rarities(id) | A fegyver ritkaságát jelzi |
| projectile\_speed | float | DEFAULT NULL | A fegyver lövedékének a sebessége |
| mag | int(11) | DEFAULT NULL | A fegyvertárnak a nagysága |
| description | text | DEFAULT ’A deadly weapon’ | A fegyver rövid leírása |
| devinfo | text | DEFAULT ’FIRE’ | Fejlesztői információk a fegyverről |

# 2.3 Normalizálási folyamat

### 0. Normálforma

bullet\_hell(**map\_id**, map\_name, map\_file\_name, map\_desc, **music\_pack\_id**, music\_pack\_name, music\_pack\_cover\_image, music\_pack\_anthem, music\_pack\_menu1, music\_pack\_menu2, music\_pack\_desc**, player\_username**, player\_points, player\_all\_wins, player\_all\_games, player\_kills, player\_deaths, player\_current\_music, player\_current\_skin, player\_password, player\_is\_admin, **weapoon\_id**, weapon\_name, weapon\_file, weapon\_damage, weapon\_fire\_rate, weapon\_semi\_auto, weapon\_rarity\_id, weapon\_rarity, weapon\_projectile\_speed, weapon\_mag, weapon\_desc, weapon\_devinfo, **player\_skin\_id**,player\_skin\_name, player\_skin\_display\_image, player\_skin\_background\_image, player\_skin\_desc)

### 1. Normálforma

players (***player\_username***, player\_points, player\_all\_wins, player\_all\_games, player\_kills, player\_deaths, player\_current\_music, player\_current\_skin, player\_password, player\_is\_admin)

bullet\_hell (***map\_id***, map\_name, map\_file\_name, map\_desc, *music\_pack\_id*, music\_pack\_name, music\_pack\_cover\_image, music\_pack\_anthem, music\_pack\_menu1, music\_pack\_menu2, music\_pack\_desc, *player\_username*, *weapoon\_id*, weapon\_name, weapon\_file, weapon\_damage, weapon\_fire\_rate, weapon\_semi\_auto, weapon\_rarity\_id, weapon\_rarity, weapon\_projectile\_speed, weapon\_mag, weapon\_desc, weapon\_devinfo, *player\_skin\_id*, player\_skin\_name, player\_skin\_display\_image, player\_skin\_background\_image, player\_skin\_desc)

### 2. Normálforma

players (***player\_username***, player\_points, player\_all\_wins, player\_all\_games, player\_kills, player\_deaths, player\_current\_music, player\_current\_skin, player\_password, player\_is\_admin)

maps (***map\_id***, map\_name, map\_file\_name, map\_desc, music\_pack\_id)

music\_packs (***music\_pack\_id***, music\_pack\_name, music\_pack\_cover\_image, music\_pack\_anthem, music\_pack\_menu1, music\_pack\_menu2, music\_pack\_desc)

weapons (***weapoon\_id*,** weapon\_name, weapon\_file, weapon\_damage, weapon\_fire\_rate, weapon\_semi\_auto, weapon\_rarity\_id, weapon\_projectile\_speed, weapon\_mag, weapon\_desc, weapon\_devinfo)

rarities (***weapon\_rarity\_id***, weapon\_rarity)

player\_skins (***player\_skin\_id***, player\_skin\_name, player\_skin\_display\_image, player\_skin\_background\_image, player\_skin\_desc)

map\_weapons (***map\_id***, *weapon\_id*)

player\_skin\_inventory (***player\_id*,** *skin\_id*)

player\_music\_inventory (***player\_id***, *music\_pack\_id*)

### 3. Normálforma

maps (***id***, name, file\_name, description, music\_pack\_id)

map\_weapons (***map\_id***, *weapon\_id*)

music\_packs (***id***, name, cover\_image, anthem, main\_menu\_theme1, main\_menu\_theme2, description)

players (***username***, points, all\_wins, all\_games\_played, kills, deaths, music\_pack\_id, active\_skin\_id) [[1]](#footnote-1)

player\_login (***username***, password, is\_admin)

player\_music\_inventory (***player\_id*, *music\_pack\_id***)

player\_skins (***id***, name, display\_image, background\_image, description)

player\_skin\_inventory (***player\_id***, *skin\_id*)

rarities (***id***, rarity)

weapons (***id***, name, file\_name, damage, fire\_rate, semi\_auto, rarity\_id, projectile\_speed, mag, description, devinfo)

## 3. Jogosultságok, jogosultsági szintek

# 3.1. Asztali Játékalkalmazás

### Jogosultsági szint: Játékos (Player)

* + **Hozzáférés:** Minden regisztrált felhasználó, aki bejelentkezik a játékba, automatikusan ezt a jogosultsági szintet kapja.
  + **Engedélyek:**
    - A játék futtatása.
    - A saját játékbeli adatok megtekintése (a players táblában tárolt adatok: pontszám, ölések, halálozások stb.).
    - A saját kiválasztott zene és karakterkinézet beállítása.
    - A saját birtokolt zenék és karakterkinézetek megtekintése (a player\_music\_inventory és player\_skin\_inventory táblák alapján).
    - A játék többi elemének (pályák, fegyverek stb.) megtekintése (de nem módosítása).
  + **Korlátozások:**
    - Más játékosok adatainak megtekintése vagy módosítása nem engedélyezett.
    - Az adatbázis közvetlen elérése és módosítása nem engedélyezett.
    - Nincs adminisztrációs jogosultság

# 3.2 Weboldal

### Jogosutlsági szint: Játékos (Player)

* + **Hozzáférés:** Minden regisztrált felhasználó, aki bejelentkezik a weboldalra, automatikusan ezt a jogosultsági szintet kapja, amennyiben az is\_admin mező értéke FALSE .
  + **Engedélyek:**
    - A saját játékbeli adatok megtekintése (a players táblában tárolt adatok, ugyanúgy, mint az asztali alkalmazásban).
    - A saját kiválasztott zene és karakterkinézet megtekintése.
    - A saját birtokolt zenék és karakterkinézetek megtekintése (a player\_music\_inventory és player\_skin\_inventory táblák alapján).
    - A játék Kódexének megtekintése (pályák, fegyverek, zenék, mechanikák, karakterkinézetek).
    - A rangsor megtekintése.
    - A saját profiloldal megtekintése és a jelszó megváltoztatása.
  + **Korlátozások:**
    - Más játékosok adatainak módosítása nem engedélyezett.
    - Az adatbázis közvetlen elérése és módosítása nem engedélyezett.
    - Nincs adminisztrációs jogosultság

### Jogosultsági Szint: Adminisztrátor (Admin)

* + **Hozzáférés:** Azok a felhasználók, akiknek a player\_login táblában az is\_admin mező értéke TRUE adminisztrátori jogosultságot kapnak a weboldalon.
  + **Engedélyek:**
    - Minden, amit a Játékos szintű felhasználó megtehet.
    - **Admin Panel elérése:** Az adminisztrációs felületen keresztül a következő műveletek végezhetők:
      * Új rekordok felvétele az adatbázisba (új pályák, fegyverek, zenei csomagok, karakterkinézetek).
      * Meglévő rekordok megtekintése és szerkesztése bármelyik táblában (maps, map\_weapons, music\_packs, players, player\_login, player\_music\_inventory, player\_skins, player\_skin\_inventory, rarities, weapons).
      * Felhasználói adatok kezelése
      * A teljes adatbázis tartalmának megtekintése.

## 4. Alkalmazások felépítése

# 4.1 Webalkalmazás

### Architektúra

A lentebbi ábra bemutatja a webalkalmazás részeit és ezek kapcsolati, interakciós ábráját.

HTTP Request

Function call

Response

Query

Response

Response

### Mappaszerkezet

* **admin**
  + **api** – Admin által elvégezhető feladatok kezelésére
  + **src** – Javascript és CSS
* **codex**
  + **src** – Javascript és CSS
* **leaderboard** – Rangsor kódja
* **login** – Bejelentkezés kódja
* **profile** – Felhasználói profil kódja
* **register** – Regisztráció kódja
* **src** - Források
  + **css**
  + **fonts** - Betűtípusok
  + **images** – Weboldalon használt képek
    - **characters**
    - **maps**
    - **music**
    - **ui**
    - **weapons**
  + **js** – Globális Javascript kód
  + **music** - zenék
  + **php** – Globális PHP fájlok

A PHP fájlok elején egy két soros require\_oncebiztosítja a kódot, hogy az adatbázis konfiguráció és egyéb segítőfüggvények be legyenek importálva.

config.php

Ez a fájl meghatározza a szervert és csatlakozik az adatbázishoz. Ha a csatlakozás nem sikerül, az oldal nem indul el.



utils.php

is\_logged\_in() és is\_admin\_logged\_in(): Ellenőrzik, hogy az admin/felhasználó be van e jelentkezve.

username\_exists() – Ellenőrzi, hogy az adott felhasználónév létezik-e az adatbázisban. Ehhez egy SQL lekérdezést futtat az adatbázisról.

is\_user\_admin() – Ellenőrzi, hogy az adott felhasználó admin-e



### Képernyőképek

### Funkciók

Bejelentkezés

**Fájlok:**

* login\_check.php
* login.css
* login.js
* login.php

login\_check.php

Egy feltétel ellenőrzi, hogy a kérés paraméterei helyesen lettek-e szolgáltatva.



A program megkapja az adatokat, és előkészít egy SQL lekérdezést. A lekérdezés védve van SQL injection-öktől. Ezután megkapja a kérés eredményét.



A program összehasonlítja a lekérdezett jelszót (hash) és a megadott jelszót (hash), és ezek alapján dönti el, hogy a bejelentkezés sikeres-e.



login.php

Megjeleníti a bejelentkezés-formot.

login.js

Ez a fájl tartalmazza a bejelentkezéshez szükséges kliensoldali kéréseket, és segítőfunkciókat.



A LoginUser() a szövegdobozba beírt jelszót hasheli (egy külső függvény segítségével), majd elküld egy kérést a login\_check.php-nak, ami visszaadja, hogy a bejelentkezés helyes-e. Ha igen, a felhasználó beléphet az index.php főöldalra, ha a bejelentkezés sikertelen, kap egy hibaüzenetet, hogy a bejelentkezés sikertelen volt.



A TogglePwVisibility() függvény a jelszómezőbe beírt jelszó láthatóságát állítja át pontozottból láthatóvá, és vissza.

login.css

Tartalmazza a bejelentkezés képernyő stílusát.

Regisztráció

A regisztráció és bejelentkezés űrlap kinézetében kevés eltérés található.

**Fájlok:**

* register\_user.php
* register.css
* register.js
* register.php

register.php

Megjeleníti a regisztrációs formot.

register\_user.php

Ez egy API az új felhasználókat regisztrálja az adatbázisba.



A kérés hitelesítése után a felhasználónév és jelszó mezőket ellenőrzi a htmlspecialchars()-val. Ezután a program ellenőrzi, hogy a felhasználónév létezik-e. Ezt a fentebb bemutatott username\_exists() metódussal teszi.

A regisztrációhoz kellően 2 SQL lekérdezést kell végrehajtani:

* players táblába beilleszteni a játékost, alapértelmezett adatokkal
* player\_login táblába beilleszteni a játékost, a bejelentkezés adataival

Sikeres beszúrás után a PHP ’username’ session változóját beállítja. Eme változóval fogja a program ellenőrizni, hogy a felhasználó be van-e jelentkezve.

register.js

A regisztrációhoz kapcsolódó segédfüggvényeket tartalmazza (jelen esetben csak egyet).



A RegisterUser() metódus API hívást indít a register\_user.php felé. Ha a regisztráció sikeres, elnavigálja a felhasználót a főoldalra.

Saját profil

Ezen az oldalon a felhasználó meg tudja tekinteni az adatait és a saját profiljához tartozó statisztikákat. Ezen az oldalon lehetséges a jelszó megváltoztatása is.

**Fájlok:**

* change\_password.js
* change\_password.php
* get\_player\_characers.php
* get\_player\_musics.php
* get\_user\_statistics.php
* index.php – az oldal megjelenítéséért felelős
* load\_profile\_data.php
* profile.css

load\_profile\_data.js

Betölti a szükséges játékos-statisztikát az oldalra:



A GetUsername() lekérdezi egy API hívással (get\_username.php) a felhasználónevet, majd ha ez kész van (async-await) meghívja a LoadUserData metódust, ami egy API hívással lekérdezi a játékos adatokat, majd ezeket megjeleníti a weboldalon.

get\_user\_statistics.php



Ez a fájl kérdezi le a játékos statisztikáit, adatait. Megkapja a pontokat, győzelmi rátát, összes lejátszott játékot, ölések, halálok és a jelenleg használt zenei csomagot is.

get\_player\_musics.php



Az adott felhasználónévhez lekérdezi az összes birtokolt zenei csomagot.

get\_player\_characters.php



Az adott felhasználónévhez lekérdezi az összes birtokolt karatkert.

change\_password.php



A megjelölt felhasználónévvel rendelkező játékos jelszavát megváltoztatja.

change\_password.js

Ez a fájl ellenőrzi a jelszóváltoztatás folyamatát. A jelszóváltoztatásnak három állapota lehetséges: Helytelen régi jelszó, Nem megegyező jelszavak és Megváltozott jelszó. A jelszó változtatásért a changePassword() felel. Az ellenőrzés folyamat a következőképp folyik:

1. isPasswordCorrect(username, password) 🡪 Hasheli a jelszót, majd ezt elküldi a login\_check.php-nak, hogy ellenőrizze a régi jelszót. Ha sikerült, továbblépünk.
2. Amennyiben a kétszer beírt új jelszó egyezik, a program küld egy kérést a change\_password.php felé.
3. Ha a kérés sikeres volt és a jelszó megváltozott, kapunk vizuális visszaigazolást erről.

# Rangsor

**Fájlok:**

* get\_leaderboard\_data.php
* index.php – a rangsor megjelenítéséért felelős
* leaderboard.css
* leaderboard.js

get\_leaderboard\_data.php

Lekéri a rangsorhoz szükséges rekordokat. A rangsor egyes soraiba a név, pontok, győzelmi ráta, játszott játékok, ölések és halálok kerülnek.

Mivel a játéknak nagy méretű adatbázist is kezelnie kell, nem kérdezhettük le egyszerre az összes játékost sorrendben. A lekérdezést a felhasználó által kiválasztható mennyiségű oldalakban kérdezzük le (pagination), így a szervernek nem kell egyszerre lekérdeznie az összes adatot, csak például 50-esével.



Itt számoljuk ki, hogy mennyivel kell eltolni a lekérdezés eredményét, hogy a helyes oldal adatait kérdezzük le.

leaderboard.js

A lekérdezett játékosokat a Player osztályban tároljuk:



A dokumentum indulásánál beállítjuk, és eseményhez kötjük a #page-size id-val ellátott select opciót. Ha a felhasználó változtatja az „egy oldalon megjelenített rekordok” méretét, ez rögtön frissül. Ezután lekérdezzük a rangsor adatait a GetLeaderboardData() metódussal:

1. Az elérhető oldalak számának (összes rekord / rekord oldalanként) lekérdezése
2. A rangsor adatainak tényleges lekérdezése, majd az adatokat egy Player típusú listába helyezése.
3. Ezután a helyezetteket külön kiemeli a program saját magunkkal együtt megkülönböztető jelzésekkel:

Az oldal alján oldal navigációs nyilak vannak elhelyezve. Ezekkel tudunk az oldalak között navigálni.

# Kódex

A kódexben 5 féle játékelemet tudunk megtekinteni. Ezeket külön, egyesével fogjuk bemutatni.

**Fájlok:**

* characters.php
* index.php
* load\_codex\_data.php
* load\_map\_weapons.php
* maps.php
* mechanics.php
* music.php
* weapons.php
* src
  + carousel\_style.css
  + characters.js
  + maps.js
  + music.js
  + style.css
  + weapons.js

#### Zene

A zenéket egy egymás alá betöltő csempékben lehet megtekinteni, illetve meghallgatni is. A zenék között keresni is lehetséges a fenti keresőmezőben.

music.js

Itt tölti be az oldal a zenét, illetve szűr a keresett zenére. Ezt a LoadMusic() és SearchForMusic() metódussal csinálja.



A zenék lejátszása közben biztosítva van, hogy a felhasználó ne tudjon egyszerre több zenét lejátszani.

A SearchForMusic() egy hasonló kérést küld mint betöltéskor, de kiegészíti egy keresési feltétellel:



Zenék lekérdezése:



A keresést minden a keresőmezőbe írt karakterkor frissíti:



Megjelenítés módszere:



#### Fegyverek

A fegyvereket egy hős-szekciós választó mechanikával lehet megtekinteni. Felül látható az éppen megtekinteni kívánt fegyver, míg alul a megtekinthető fegyverek. A fegyverek kezelésére a Weapon osztály szolgál:



A LoadWeapons() lekérdezi a fegyvereket egy API hívással:



Miután a Weapon típusú listába feltöltöttük a fegyvereket, meghívjuk a DisplayWeapons() metódust. Ez megjeleníti a kicsi csempéken a fegyvereket, illetve gondoskodik a hover animációkon is.

DisplayHeroWeapon(): A jelenleg megtekintett fegyver indexjét egy dokumentumszintű változóban tároljuk, és ezt jelenítjük meg nagyban.

#### Pályák

A pályákat egy irányítható képnézegetőben lehet megtekinteni. Minden pályához tartozik egy név és leírás. A pályákat egy BHMap nevű osztályban tároljuk:



A betöltési logika hasonlóan működik, mint a fegyvereknél. A LoadMaps() betölti a BHMap típusú listába az adatokat, majd a DisplayMaps() megjeleníti a képnézegetőben.

A képnézegető vezérléséhez külön logikát kellett alkalmazzunk:



A navigációs nyilak lenyomásakor a slide() funkció meghívásakor a kép a helyes irányba csúszik. Ezt animációs osztályok alkalmazásával tettük lehetővé.

Ezen felül a háttér képét is változtatjuk, ha a felhasználó kis képernyőjű eszközön használja az oldalt.

#### Karakterek

A karakterek egy különleges kialakításban jelennek meg, minden karakter elfoglal egy sort és az egér ráhúzásával tudhatunk meg róluk több információt.

A karakterek tárolására a Character osztály szolgál:



Itt is a Load-Display megoldást alkalmaztuk. A LoadCharacters() metódus betölti a karaktereket:



A DisplayCharacters() a karaktersávokat tölti be és jeleníti meg, a hozzájuk tartozó animációkkal:



#### Mechanikák

A mechanikák oldal a bemutatásra szoruló mechanikákat írja le és összegzi. Adatbázisban való jelenlét hiányában ez a funkció nem kommunikál az adatbázissal.

#### load\_codex\_data.php

Ez a fájl felelős az összes kódex adat betöltéséért. A ’data\_type’ GET paraméternek köszönthetően eldönthető, hogy a kérés ’maps’,’characters’,’music’ vagy ’weapons’ típusú, és ezekhez megfelelő lekérdezéseket futtat:



#### load\_map\_weapons.php



Ez a fájl a GET paraméterben megadott map\_id-jű pályához kérdezi le az összes hozzárendelt fegyvert (Azaz azokat a fegyvereket, amelyek a pályán előfordulhatnak).

# Admin panel

**Fájlok:**

* admin\_header.php
* admin\_login\_check.php
* admin\_login.php
* character\_management.php
* index.php
* map\_management.php
* music\_management.php
* user\_management.php
* weapon\_management.php
* api
  + admin\_add\_new\_character.php
  + admin\_delete\_user.php
  + admin\_add\_new\_map.php
  + admin\_delete\_weapon.php
  + admin\_add\_new\_music.php
  + admin\_get\_character\_data.php
  + admin\_add\_new\_weapon.php
  + admin\_get\_map\_data.php
  + admin\_delete\_character.php
  + admin\_get\_music\_data.php
  + admin\_delete\_map.php
  + admin\_get\_user\_data.php
  + admin\_delete\_music.php
  + admin\_get\_weapon\_data.php
* src
  + character\_management.js
  + script.js
  + weapon\_management.js
  + map\_management.js
  + style.css
  + music\_management.js
  + user\_management.js

Az adminfelület 7 oldalból áll, melyből 5 a játékelemek manipulálására szolgál.

index.php és admin\_header.php:

Az index.php az adminpanel főoldalja, itt nem találunk több információt. Az admin\_header a weboldal többi részéhez használt header.php-hoz hasonlítható, melyet minden megjelenítendő oldalon meghívunk a require\_once() php függvénnyel.

admin\_login.php

Minden admin oldal kezdetén egy ellenőrzés látható, ami nem engedi tovább a felhasználót, ha nincs bejelentkezve:



Ez a fájl az admin bejelentkezéséhez szükséges jelszót kérdezi meg a felületre való átlépés előtt mégegyszer.

A script.js-ben megtalálható LoginAdmin() metódussal teszi ezt:



admin\_login\_check.php

A kapott felhasználónév és jelszóval ellenőrzi, hogy létezik-e ilyen felhasználó, akinek admin jogosultsága van.

Az admin feladatokat ellátó API-nak hasonló felépítése van egymáshoz. A hozzáadás, törlés és lekérés műveleteket különböző táblákon, de hasonló módszerekkel végzik.

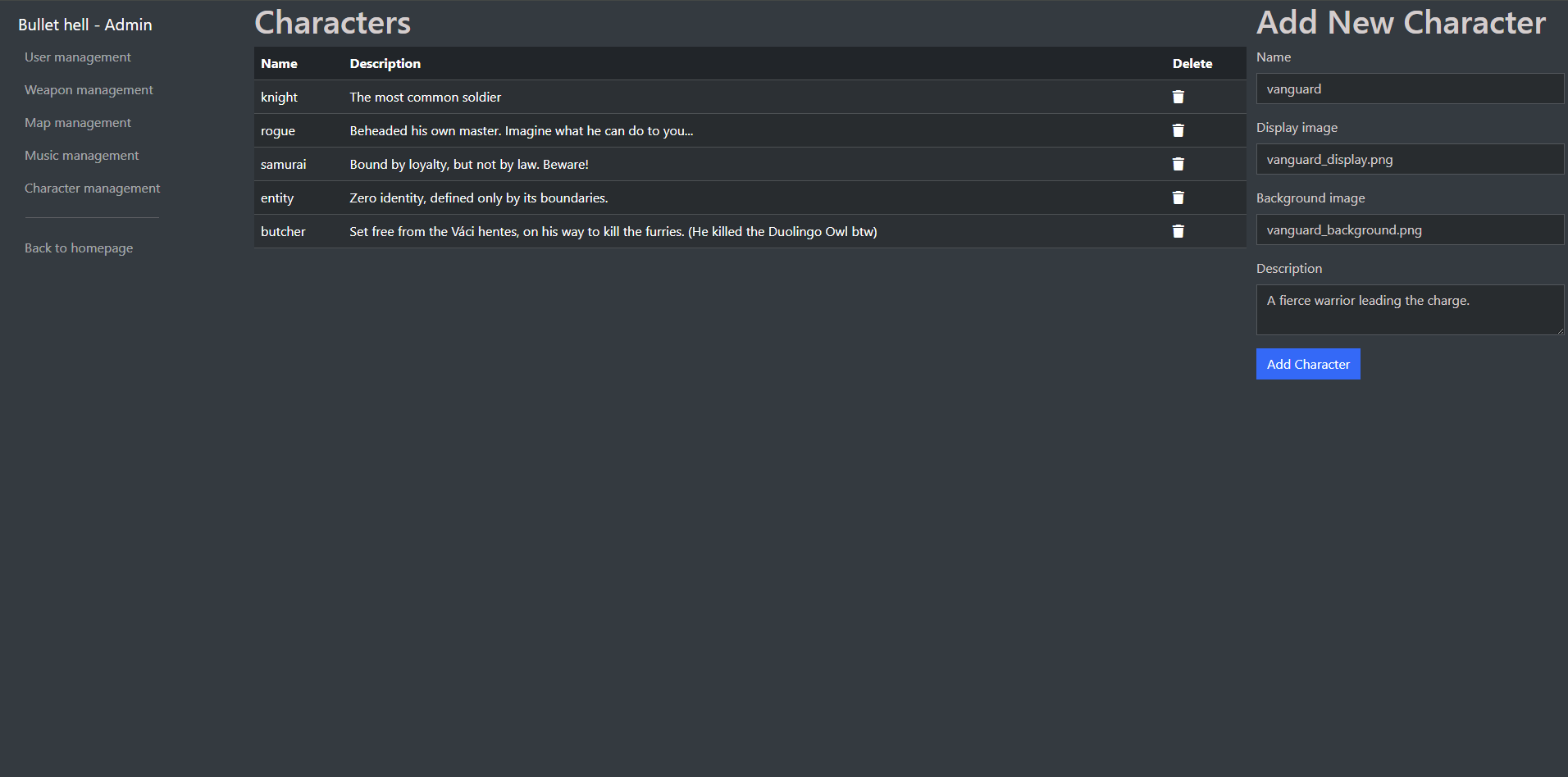
#### Karakter menedzsment

**Fájlok:**

* character\_management.php
* admin\_add\_new\_character.php
* admin\_delete\_character.php
* admin\_get\_character\_data.php

character\_management.php

Megjeleníti a karakterek táblázatot egy felviteli formmal mellette.



admin\_add\_new\_character.php

Karakter beszúrása az adatbázisba:



admin\_delete\_character.php



DELETE típusú kéréssel kitörli az adott ID-jű karaktert.

admin\_get\_character\_data.php



Lekérdezi az összes karakter adatát.

character\_management.js

A karakterek táblázatát és felviteli formját kezeli. A LoadCharacterTable() metódus az admin\_get\_character\_data.php-tól kérdezi le a karaktereket, majd jeleníti meg a táblázatban.

A DeleteCharacter(id) metódus kérést küld az admin\_delete\_character.php-nak, hogy törölje ki a megadott id-jű karaktert.



A dokumentum készenlétekor eseményfigyelőt rakunk az új karakter hozzáadása formhoz, hogy annak tüzelésekor hozzáadjuk az új karaktert az adatbázishoz.

#### Fegyver menedzsment

**Fájlok:**

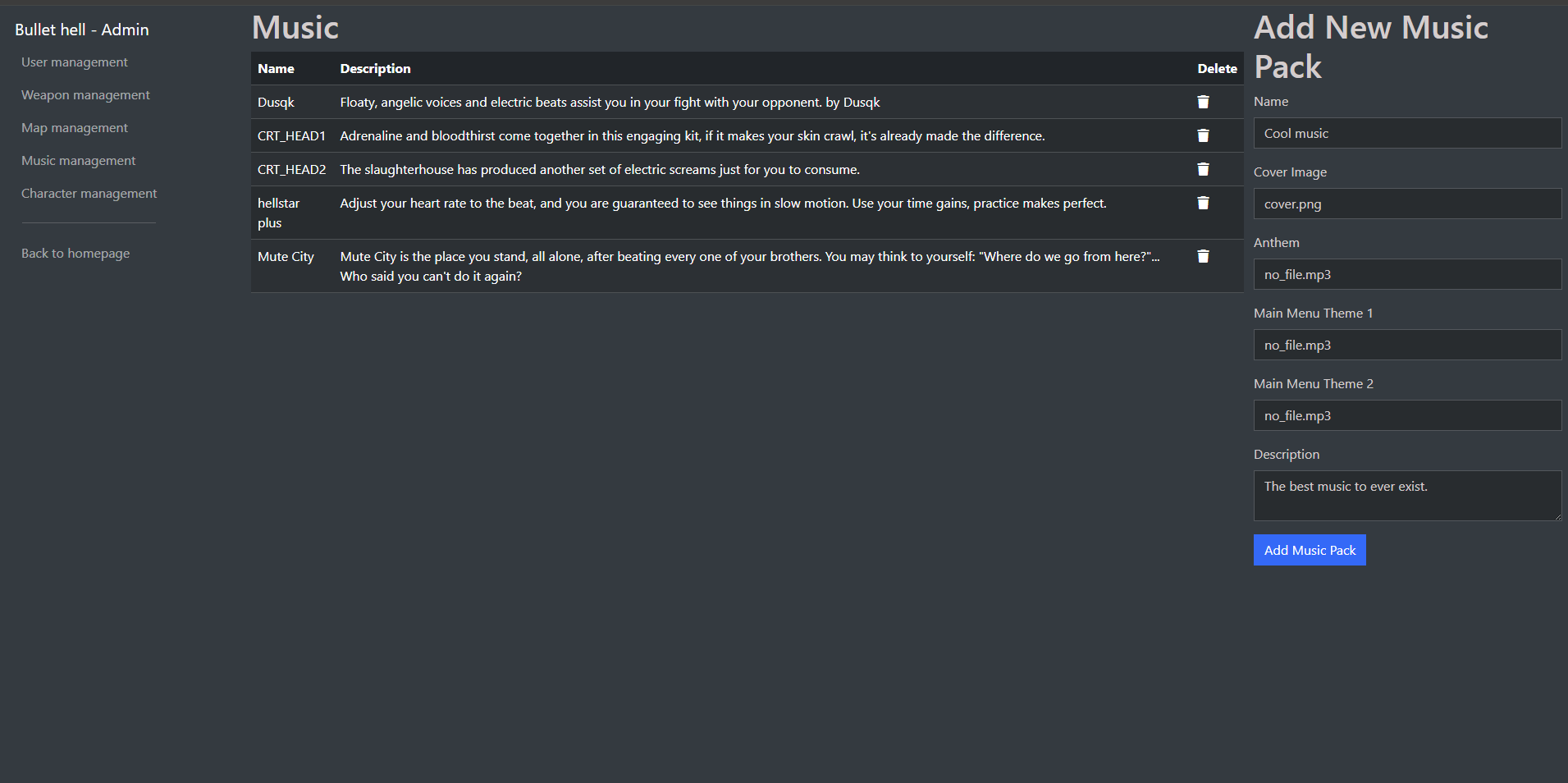
* weapon\_management.php - Megjeleníti a fegyverek táblázatot egy felviteli formmal mellette.
* weapon\_management.js
* admin\_add\_new\_weapon.php – Új fegyver felvétele
* admin\_delete\_weapon.php – Fegyver törlése id alapján
* admin\_get\_weapon\_data.php – Összes fegyver betöltése



#### Zene menedzsment

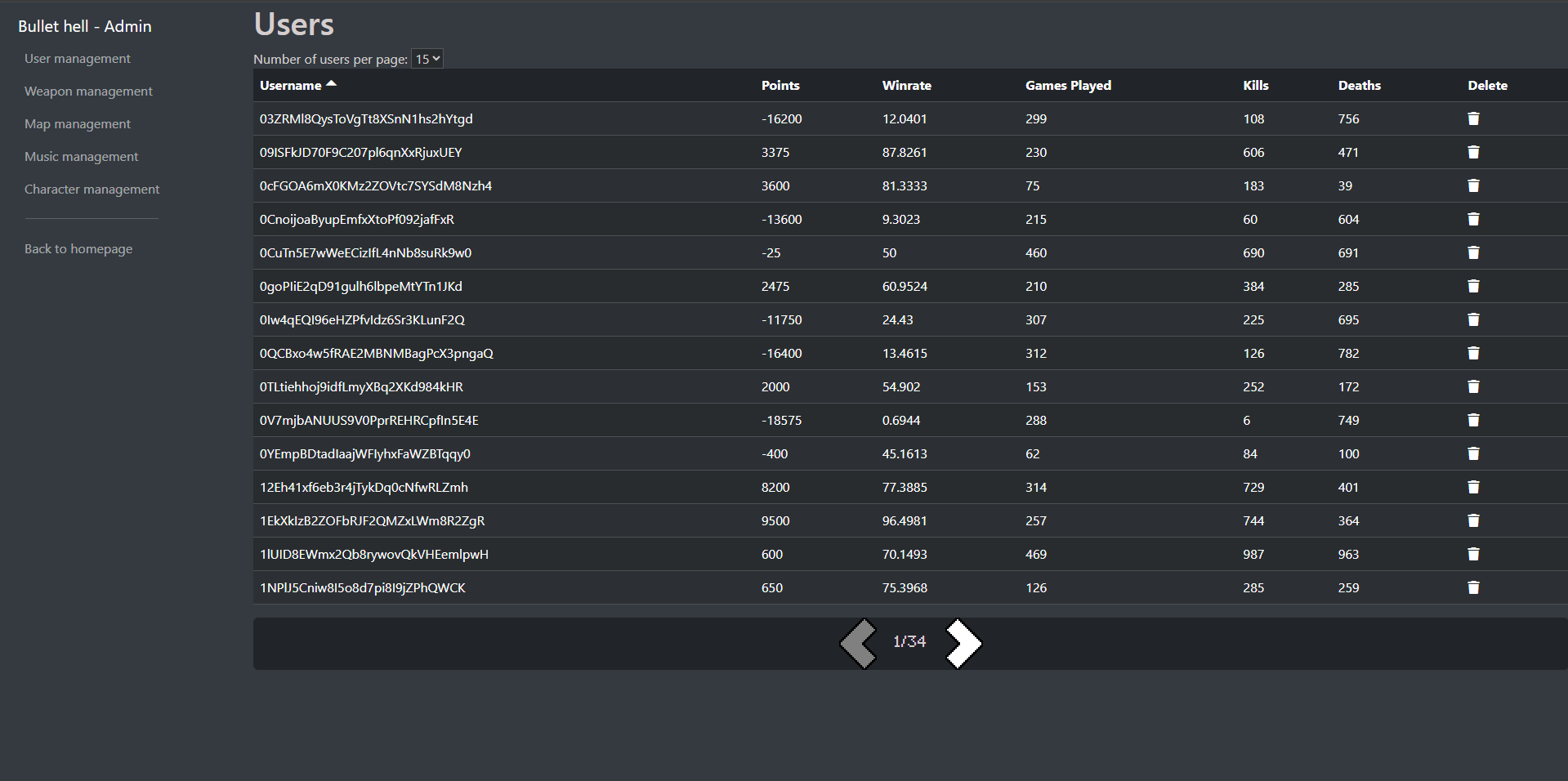
**Fájlok:**

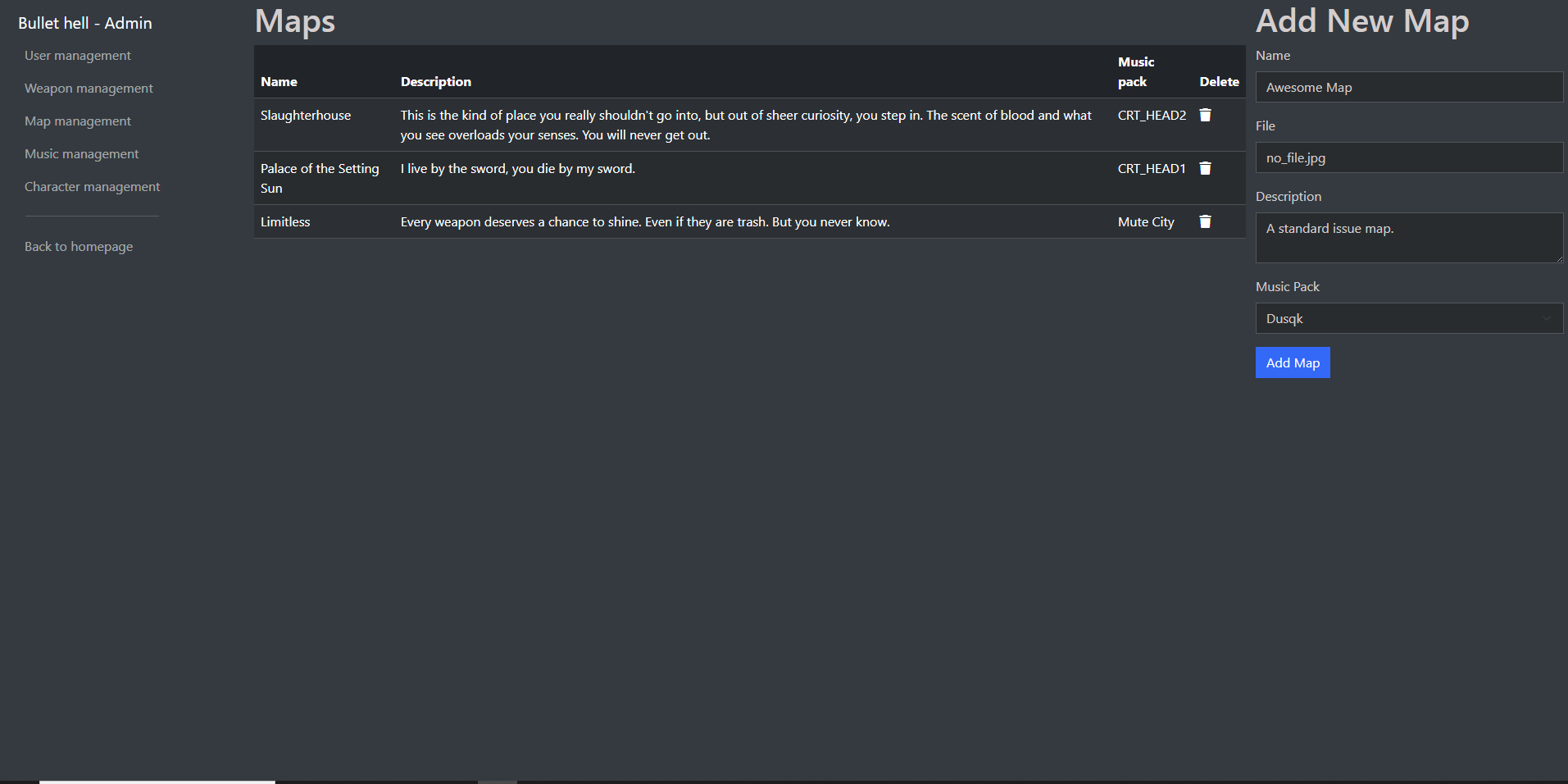
* music\_management.php – Megjeleníti a zenék táblázatot, mellette egy felviteli formot.
* music\_management.js – Táblázat feltöltése adatbázisból, törlés, hozzáadás kezelése
* admin\_add\_new\_music.php – Új zene felvétele
* admin\_delete\_music.php – Zene törlése ID alapján
* admin\_get\_music\_data.php – Az összes zene lekérdezése



#### Pálya menedzsment

**Fájlok:**

* map\_management.php – Megjeleníti a pályák táblázatot, mellette egy felviteli formot.
* map\_management.js – Táblázat feltöltése adatbázisból, törlés, hozzáadás kezelése
* admin\_add\_new\_map.php – Új pálya felvétele az adatbázisba
* admin\_delete\_map.php – Pálya törlése ID alapján
* admin\_get\_map\_data.php – Az összes pálya lekérdezése
* 



#### Felhasználó menedzsment

* user\_management.php – Megjeleníti a táblázatot.
* user\_management.js – Táblázat feltöltése adatbázisból, törlés kezelése
* admin\_delete\_user.php – Felhasználó törlése ID alapján
* admin\_get\_user\_data.php – Az összes felhasználó lekérdezése

### Input adatok helyességének ellenőrzése

#### Webes alkalmazás

Az webes alkalmazás biztonsága és az adatintegritás megőrzése érdekében kiemelt figyelmet fordítottunk a felhasználói input adatok helyességének és ártalmatlanságának ellenőrzésére. A validáció két fő szinten valósul meg: ***kliensoldalon*** a felhasználói élmény javítása érdekében, és ***szerveroldalon*** a biztonság és az adatok egységességének garantálása céljából.

1. **Kliensoldali validáció**

A kliensoldali validáció célja az azonnali visszajelzés biztosítása a felhasználó számára a helytelenül kitöltött űrlapmezőkről, csökkentve ezzel a felesleges szerveroldali terhelést.

Itt elsősorban a HTML5 beépített validációs eszközeit használtuk.

Példa:



1. **Szerveroldali validáció**

Minden, a kliensoldalról érkező adatot kötelezően ellenőrzünk mielőtt bármilyen műveletet végzünk velük az adatbázisban.

**Alapvető ellenőrzések**

* Az *isset()* függvénnyel ellenőrizzük, hogy a várt $\_POST és $\_GET paraméterek valóban léteznek-e.
* Az empty() függvénnyel biztosítjuk, hogy a kötelező mezők ne legyenek üresek

**Biztonsági Intézkedések**

* **SQL Injection megelőzése**
  + Minden adatbázis lekérdezés során prepared statementeket használunk. Ez a módszer megakadályozza, hogy a felhasználótól érkező kódok esetleg rosszindulatú SQL parancsokként legyenek értelmezve.
* **Példa:**

****

* **XSS (Cross-Site scripting) megelőzése**
  + Minden olyan felhasználói inputot, amely potenciálisan megjelenítésre kerülhet a HTML oldalon, a *htmlspecialchars()* függvénnyel kezelünk. Ez a függvény átalakítja a speciális HTML karaktereket azok HTML entitás megfelelőire így megakadályozza, hogy a böngésző ezeket HTML kódként vagy JavaScriptként értelmezze.
* **Példa:**

****

1. *doesnt technically have to be two separate tables but it makes sense to not mix login info with gameplay data* [↑](#footnote-ref-1)