## 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

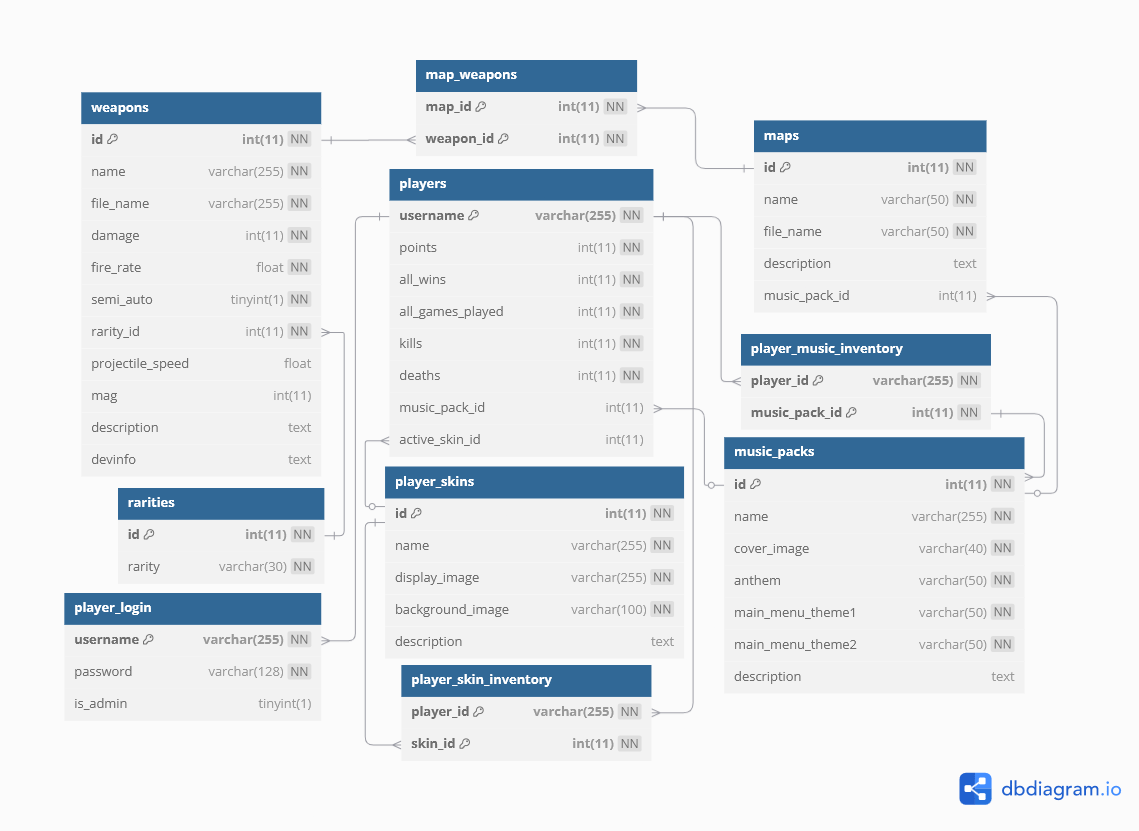
# 1.3 Fejlesztői eszközök

* **Böngésző:** Google Chrome, Mozilla Firefox
  + A weboldal teszteléséhez a fentebb említett böngészőkön teszteltük.
* **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, **weapon\_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*, *weapon\_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 (***weapon\_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

### Jogosultsági 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 törlése maps, weapons, music\_packs, players és player\_skins táblákban.
      * 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.

Response

Response

Query

Function call

Response

HTTP Request

### 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 a felhasználó/admin 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



### 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 karaktert.

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 még egyszer.

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.

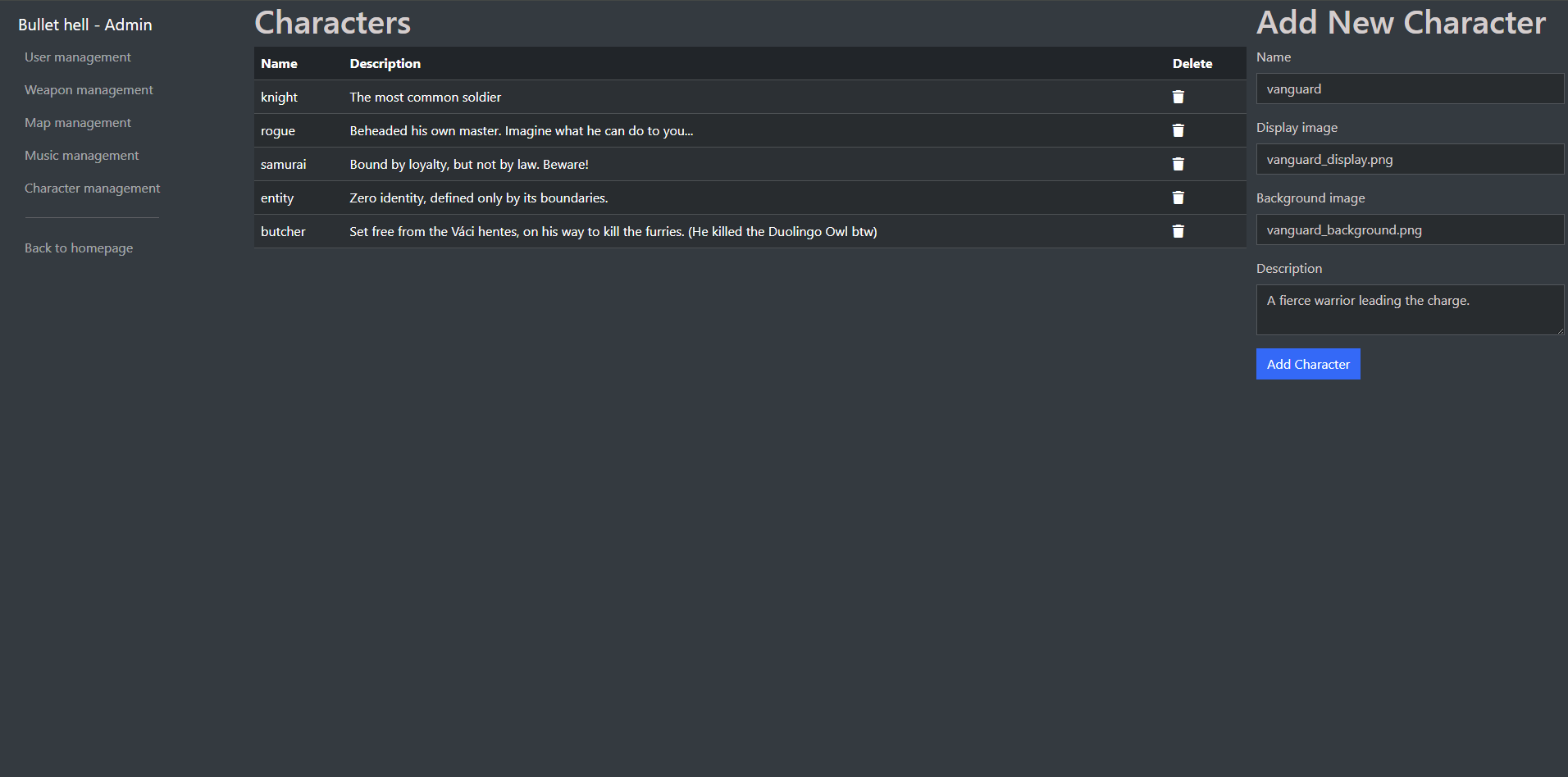
#### Karakter menedzsment

**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:**

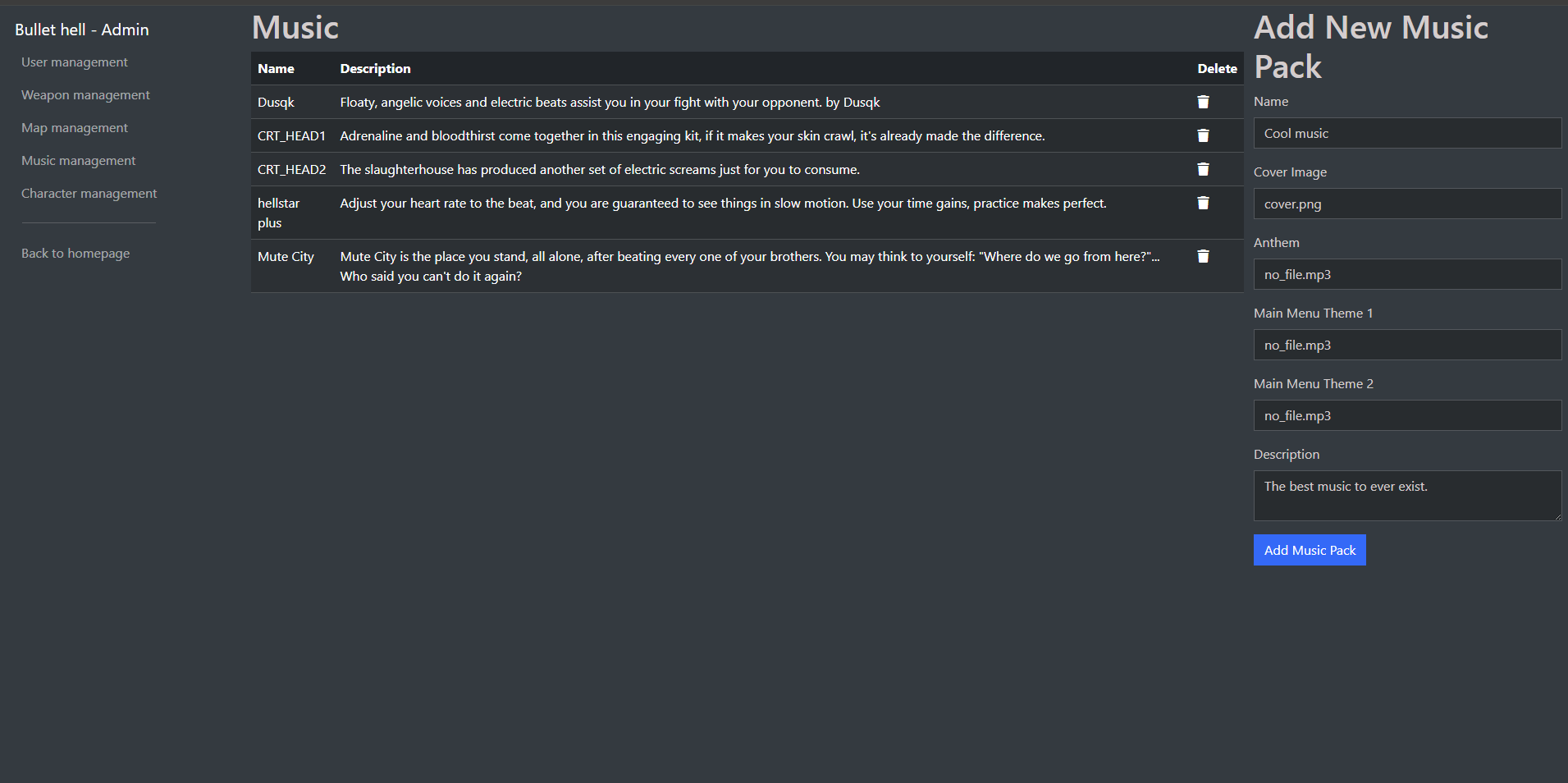
* weapon\_management.php - Megjeleníti a fegyverek táblázatot egy felviteli formmal mellette.
* weapon\_management.js
* admin\_add\_new\_weapon.php – Új fegyver felvétele
* admin\_delete\_weapon.php – Fegyver törlése id alapján
* admin\_get\_weapon\_data.php – Összes fegyver betöltése



#### 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 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 megakadályozva, hogy a böngésző ezeket HTML kódként vagy JavaScriptként értelmezze.
* **Példa:**

****

# 4.2 Asztali alkalmazás

### 4.2.1 Áttekintés

Ez a dokumentum a *Bullet Hell* projekt asztali játékalkalmazásának technikai részleteit tárgyalja. A játék egy 2D, 1v1 lokális többjátékos platformer lövöldözős játék, amelyet PC platformra fejlesztettünk. A játékmenet középpontjában a gyors tempójú, ügyességalapú küzdelem áll, amely több különböző pályán zajlik, széles fegyverválaszték használatával. A játék integrálódik egy webes háttérrendszerrel a játékosok fiókkezelése és a rangsor nyilvántartása céljából.

### 4.2.2 Technológiák

* **Játékmotor:** Unity Engine
* **Programozási nyelv:** C#
* **Integrált fejlesztői környezet (IDE):** Visual Studio
* **Verziókezelés:** Git, Github

### 4.2.3 Projekt struktúra

Assets/

├── FX/ # Vizuális effektusokat és kapcsolódó asseteket tartalmaz.

├── levels/ # Pálya-specifikus assetek helye.

├── menu/ # A játék menürendszereihez tartozó UI elemek és logika.

├── Music/ # A játék zenei anyagai

├── player/ # A játékos karakterekhez kapcsolódó minden asset

├── Plugins/ # Külső bővítmények, library-k és importált csomagok.

├── Presets/ # Előre definiált beállítások Unity komponensekhez

├── Scenes/ # A projekt összes Unity jelenetét, mint a menük és a játékpályák tartalmazza.

├── scripts/ # A játék logikáját megvalósító összes C# script központi helye

├── splashes/ # A játék indításakor megjelenő képernyők assetjei.

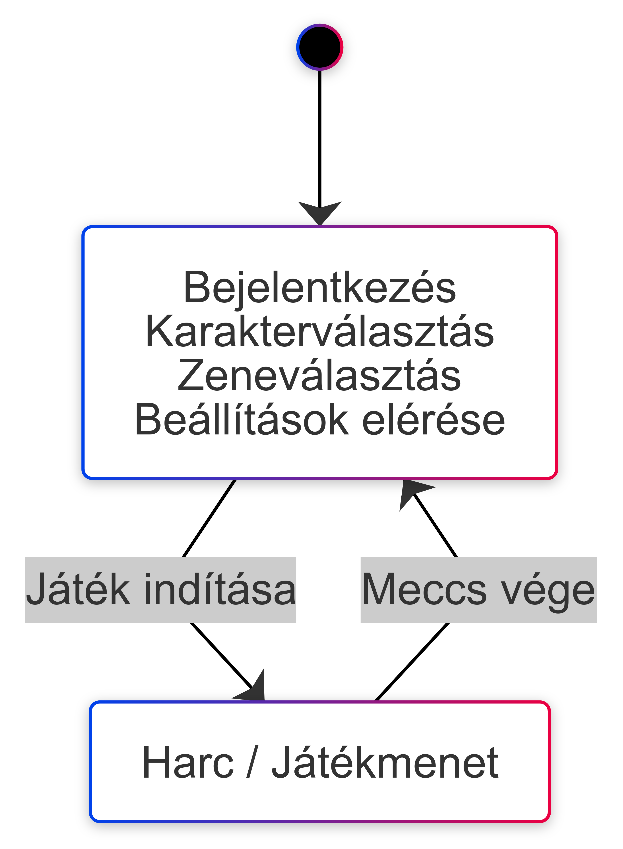
├── UI/ # Általános felhasználói felületi elemek és kapcsolódó assetek.

└── weapons/ # A játékban szereplő fegyverekhez kapcsolódó assetek.

### Projekt architektúra

Ez a diagram a játék szoftverének fő komponenseit és azok alapvető kommunikációs kapcsolatait mutatja be magas szinten. A rendszerek közötti nyilak az adat vagy vezérlés áramlásának irányát jelzik.

1. **Input Rendszer (Input):** Ez a komponens felelős a játékos által használt beviteli eszközök (billentyűzet, egér, gamepad) jeleinek fogadásáért és feldolgozásáért.
   * **Input -> UI:** Az Input Rendszer továbbítja azokat a bemeneti jeleket (pl. egérkattintás, egérmozgás, kontroller joystick mozgatás), amelyek a felhasználói felület elemeivel (gombok, beviteli mezők) való interakcióhoz szükségesek.
   * **Input -> Game Manager:** Az Input Rendszer továbbítja a játékmenet szempontjából releváns parancsokat (pl. karakter mozgása, ugrás, lövés) a Game Manager felé feldolgozásra.
2. **Felhasználói Felület (UI):** Ez a rendszer felelős a játékos számára megjelenő grafikus elemek (menük, HUD, gombok, szövegek) megjelenítéséért és az azokon keresztüli interakciók kezeléséért.
   * **UI -> Game Manager:** A felhasználói felületen végrehajtott akciók (pl. "Fight" gomb megnyomása) parancsokat küldenek a Game Managernek a megfelelő játéklogikai folyamatok elindítására.
   * **UI -> API Kezelő:** Amikor a felhasználó olyan műveletet végez, amely szerveroldali kommunikációt igényel (pl. bejelentkezés, megszerzett zsákmány lekérése), a UI továbbítja a kérést az API Kezelő felé.
3. **Game Manager (GM):** A játék központi agya, amely a fő játéklogikát, a játékszabályokat, a meccsek állapotát és a játékosok interakcióit kezeli.
   * **GM -> UI:** A Game Manager adatokat küld a Felhasználói Felületnek annak érdekében, hogy az naprakész információkat jelenítsen meg a játékos számára (pl. aktuális életerő, pontszám, hátralévő idő).
   * **GM -> Vizuális Rendszer / SFX:** A játékmenet során bekövetkező események (pl. lövés, találat) alapján a Game Manager jelzéseket küld a Vizuális Rendszernek a megfelelő grafikai effektek (animációk, részecskeeffektek) és hanghatások (SFX) elindítására.
   * **GM -> API Kezelő:** A játékmenet fontos eseményei után (pl. meccs befejezése) a Game Manager elküldi a releváns adatokat (eredmények, statisztikák) az API Kezelőnek, hogy azok a szerveren tárolásra kerüljenek.
4. **API Kezelő (API):** Ez a komponens felelős a játék kliens és a backend szerver (weboldal adatbázisa) közötti kommunikációért.
   * **API -> UI:** A szervertől kapott válaszokat és adatokat (pl. bejelentkezés sikeressége/sikertelensége, lekérdezett ranglista adatok, felhasználói profil információi) az API Kezelő továbbítja a Felhasználói Felület felé megjelenítésre.
   * **API -> Game Manager:** A szervertől kapott adatok (a játékos pontszáma, feloldott tartalmak) közvetlenül a Game Manager számára is relevánsak a játékmenet inicializálásához.
5. **Vizuális Rendszer / SFX (Visuals):** Ez a rendszer felelős a játék grafikus elemeinek rendereléséért, animációkért, speciális effektekért és a hanghatások lejátszásáért a Game Manager utasításai alapján.

A játék alapvető működése egy egyszerű állapotgéppel modellezhető, amely két fő állapot között tesz különbséget: a Menü és a Harc. A játékos ezek között az állapotok között mozog a játék használata során. Az alábbi ábra vizuálisan is szemlélteti ezt a folyamatot.

### 4.2.4 Rendszerek bemutatása

Input rendszer

1. **playerControls.cs**: Definiálja a játékos által használható input akciókat és azok hozzárendelését a konkrét beviteli eszközökhöz (billentyűzet, egér és gamepad).
2. **playerController.cs**: Közvetlenül felhasználja a playerControls.cs által biztosított input akciókat a játékos karakterének mozgatására (balra/jobbra haladás, ugrás) és más alapvető, mozgáshoz kapcsolódó akciók kiváltására.
3. **menu.cs**: Kezeli a főmenü és annak almenüinek interaktív elemeire (gombok, csúszkák stb.) érkező felhasználói inputokat (pl. egérkattintásokat) a menüben való navigáláshoz és funkciók eléréséhez.
4. **loginFormNavigation.cs**: Specifikusan a bejelentkezési képernyő input mezői és gombjai közötti navigációt és interakciót (pl. Tab-bal lépkedés, Enterrel küldés) kezeli a felhasználói bevitel alapján.

##### Mozgás kiszámítása

A unity által megadott fix időközönként kiszámítja a szükséges mozgás irányát, a mászást és az irányt amerre a játékos néz, és aszerint cselekszik.

##### Tüzelés, alternatív tüzelés

Tüzeléskor a Fire() metódust hívjuk meg, ami meghívja a játékos kezében tartott fegyver Fire() metódusát. Ez a metódus a weapon.cs virtuális metódusa, ami minden fegyvernél egyedi.

##### Bejelentkezés, navigáció

A menü „Login” gombja lenyomásakor felugrik a bejelentkezés ablak. Itt két szövegmező található. A szövegmezők között Tab-bal lehet lépkedni, és az enter gomb lenyomásával továbblépni. Ezt a loginFormNavigation.cs teszi lehetővé:



Felhasználói felület

1. **menu.cs**: A játék főmenüjének és almenüinek (pl. karakterválasztás, beállítások a játékban) központi logikáját és működését valósítja meg, kezeli a gombnyomásokat és a menüpontok közötti navigációt.
2. **loginFormNavigation.cs**: Kifejezetten a bejelentkezési képernyő felhasználói felületének elemeit (beviteli mezők, gombok) és az azokon belüli interakciókat kezeli.
3. **fightUi.cs**: Felelős a harc közbeni felhasználói felület elemeinek megjelenítéséért és dinamikus frissítéséért (játékosok nevei, pontszám, hátralévő idő).
4. **healthbar.cs**: Specifikusan a játékosok életerejét vizuálisan megjelenítő életerőcsíkok működését és frissítését kezeli a HUD-on vagy a karakterek felett.
5. **crosshair.cs**: A célkereszt megjelenítését és viselkedését (pozícióját az egérhez képest) menedzseli a játék képernyőjén.
6. **menuRefs.cs**: Egy gyűjtőosztály a menü referenciái számára
7. **musicShowFight.cs**: A harc elején UI-n megjeleníti az aktuálisan játszott zene adatait (cím, előadó).
8. **musicShowMenu.cs**: A menü felületén jeleníti meg az aktuálisan játszott zene adatait.
9. **displaySkin.cs**: A karakterváltáshoz tölti be a megfelelő asseteket.

##### Bejelentkezés



A „loginPlayer()” függvény a megkapott felhasználónévvel és jelszóval bejelentkezési kérést küld az APIManager-nek, amely ellenőrzi az adatokat, és ha helyes, a loginSuccess(), ha helytelen a loginFailure függvényt hívja meg.

A sikeres bejelentkezés után a ellenőrizzük, hogy player1 vagy player2-jelentkezett-e be. Ezután beállítjuk a szükséges adatokat, referenciákat, majd újabb API kérést indítunk a birtokolt zene és karakterek irányába. Ezek sikeres lekérése után a LoadOwnedMusic() illetve a LoadOwnedCharacters() hívódik meg.

LoadOwnedMusic (a LoadOwnedCharacters ugyanerre a mintára íródott):



##### Menü

* Kezeli a bejelentkezés funkciót (lásd feljebb)
* customUpdate() metódus
  + Frissíti az aktuálisan aktív játékoshoz (activePlayer alapján, ami "p1" vagy "p2" lehet) tartozó kinézetet, nevet és zenei csomagot.
* menuUpdate() metódus
  + Ellenőrzi, hogy mindkét játékos be van-e jelentkezve. Ha igen, akkor a harc indítására szolgáló gombot kattinthatóvá teszi. Emellett ellenőrzi, hogy a játékosok nevei eltérnek-e az alapértelmezettől (jelezve a bejelentkezést), és ha igen, frissíti a menüben látható karaktermodelljeik kinézetét a kiválasztott kinézetnek megfelelően.
* Karakterválasztó metódusok
  + Kezelik egy-egy specifikus karakterkinézet kiválasztását az aktív játékos számára.
* Zeneválasztó metódusok
  + Kezelik egy-egy specifikus zenecsomag (music kit) kiválasztását az aktív játékos számára.
* reshuffle()
  + Újrakeveri a zenei csomagban szereplő zenék sorrendjét, és elindítja a lejátszást
* settingScreen()
  + Átvált a beállítások képernyőre
* musicShowMenu.cs
  + Egy beúszás – kiúszás animációval jeleníti meg az aktuálisan játszott zenét.

###### Menü Választó Csempék Kezelője – tileManager.cs

Kezeli a skin- és zeneválasztó csempék vizuális állapotát (alap/kiválasztott kép) és interaktivitását (engedélyezett/letiltott) a menüben, az alapján, hogy az aktív játékos mely elemeket birtokolja és melyiket választotta ki aktuálisan. Frissíti a központi passedData objektumot a választásokkal.Ez a kódrészlet „engedélyezi” avagy interaktálhatóvá teszi a játékos által birtokolt csempéket.

##### Harc közben

###### Életerősáv kezelése – healthbar.cs

Ez a szkript jeleníti meg az adott játékos életerősávját, illetve kezeli a életerő változását.



###### Célkereszt kezelése – crosshair.cs

Ez a rendkívül egyszerű szkript biztosítja, hogy a célkereszt objektum mindig a szülő objektumának pozícióját kövesse.

Ez a kódrészlet frissíti a célkereszt pozícióját minden képkockában.

###### Harc közbeni zeneszám információ megjelenítése – musicShowFight.cs

Ez a szkript felelős azért, hogy a harc közben éppen játszott zeneszám információi (borítókép, cím, előadó) megjelenjenek a képernyőn egy rövid animációval (beúszás, várakozás, kiúszás).



Az updateShow() a fő belépési pont az megjelenés frissítésére. A musicPlayer.cs playMusic() metódusa hívja meg a harc kezdetekor.

###### Harc közbeni HUD elemeinek megjelenítése – fightUi.cs

Ez a szkript a megkapott játékosadatok alapján beállítja a játék HUD (Heads Up Display)-ját: A játékos neve, pontja és az éppen használt fegyvert (mindkét játékos esetében).



Játéklogika – Game Manager

1. **gameManager.cs:** A játékmenet központi vezérlője. Felelős a meccs elindításáért, befejezéséért, a körök kezeléséért, a győztes meghatározásáért és a játék általános állapotának menedzseléséért a harc során.
2. **playerController.cs:** Fogadja a feldolgozott input jeleket (az Input Rendszertől) és ez alapján irányítja a játékos karakterének fizikai mozgását.
3. **groundCheck.cs:** Egy segédkomponens, amely ellenőrzi, hogy a játékos karakter a talajon tartózkodik-e. Ezt az információt a playerController.cs használja fel az ugrás logikájához.
4. **playerAnim.cs:** A játékos karakterének animációit vezérli. A playerController, playerHealth, és gunHolder állapotai alapján váltogatja a megfelelő animációkat (pl. futás, ugrás, lövés, sebződés, halál).
5. **playerHealth.cs:** Kezeli a játékos életerejét. Nyilvántartja az aktuális életerő pontokat, feldolgozza a beérkező sebzést (pl. lövedék találatból), és kezeli a karakter halálának eseményét.
6. **aim.cs:** A játékos célzási logikáját valósítja meg. Az egér kurzor pozícióját követve beállítja a karakter vagy a fegyver forgását, meghatározva ezzel a lövés irányát.
7. **gunHolder.cs:** Ez a központi szkript, amely a játékos karakteréhez csatolva kezeli az aktuálisan használt fegyvert. Felelős a fegyver vizuális megjelenítéséért, a fegyver felvételéért, eldobásáért, a célzás irányának beállításáért, a karakter nézési irányához igazodó vizuális tükrözésért, a tüzelési parancsok továbbításáért a fegyver saját szkriptjének, és a kapcsolódó UI elemek frissítéséért.
8. **weapon.cs:** Ez egy absztrakt osztály, ami a fegyverek alapvető működését definiálja (pl. lövés metódus, tárkapacitás, újratöltési idő). A specifikus fegyver szkriptek ebből származhatnak vagy ezt használhatják komponensként.
9. **weapons (mappa):** Ez a mappa tartalmazza az egyes konkrét fegyvertípusokhoz (pl. pisztoly, muskéta, láncfűrész) tartozó specifikus szkripteket, amelyek megvalósítják az adott fegyver egyedi viselkedését, tüzelési módját, sebzését stb. (a weapon.cs-re épülve).
10. **bullet.cs:** A kilőtt lövedék (projectile) viselkedését írja le. Ez magában foglalja az ütközésérzékelését, és azt, hogy mi történik találat esetén (sebzés okozása, megsemmisülés).
11. **meleeCheck.cs:** Ez a komponens egy Collider2D (trigger) segítségével érzékeli, hogy egy közelharci támadás során melyik ellenfél tartózkodik a támadás hatókörében.
12. **mapLoader.cs:** Felelős a kiválasztott pálya betöltéséért a harc jelenetbe.
13. **tileManager.cs:** A menüben megjelenítendő karakter és zene csempékért felel.
14. **spawnPositions.cs:** Definiálja és kezeli az egyes pályákon található fegyver spawnpointokat ritkaság alapján.
15. **weaponLoader.cs:** A játékban elérhető fegyverek adatainak vagy prefabjainak betöltéséért felelős, hogy a gunHolder és más rendszerek hozzáférjenek a szükséges fegyverinformációkhoz.
16. **lootGiver.cs:** A meccs végén, az eredmények alapján kiosztja a jutalmakat a játékosoknak (pl. pontok, új kinézetek feloldása).
17. **scanner.cs:** Ez a szkript felelős a játékos karakter (vagy bármely objektum, amelyre rá van helyezve) közelében lévő, felvehető tárgyak (specifikusan a "weapon" címkével ellátott GameObject-ek) érzékeléséért egy 2D Trigger Collider segítségével.

##### gameManager.cs

A gameManager.cs felelős a játék egészének irányításáért. Ebbe beletartozik a csata jelenet menedzselése, valamint a menü és a harc közötti átmenetek kezelése.

* **Inicializácó:** Az Awake metódus a szkript betöltődésekor fut le, és az aktuálisan aktív jelenet ("menu" vagy "fight") alapján dönti el, melyik inicializáló függvényt hívja meg (initMenu vagy initFight). Beállítja, hogy a játék már nem az első betöltésnél tart (passedData.firstLoad = false).
* **Harc és Kör Menedzsment:** 
  + A FixedUpdate() függvény rendszeres időközönként ellenőrzi a harc állapotát.
  + roundEnd(): Meghatározza, hogy egy adott kör véget ért-e. Ez bekövetkezhet, ha az egyik játékos meghal (playerHealth.dead), vagy ha lejár a köridő (fightUi.timeLeft < 0) és az egyik játékosnak több életereje van. Ha egy kör véget ér, növeli a győztes játékos pontszámát (p1Wins vagy p2Wins), megsemmisíti a még repülő lövedékeket, és true értékkel tér vissza.
  + fightEnd(): Ellenőrzi, hogy a teljes mérkőzés befejeződött-e. Ez akkor történik meg, ha valamelyik játékos eléri a győzelemhez szükséges körök számát (2 győzelem, "Best of 3" logika). Beállítja a winner változót, és true-val tér vissza.
* **Kör vége, harc visszaállítása**
  + resetFight(): Előkészíti a pályát és a játékosokat egy új kör kezdetére. Frissíti a harci UI-t (fightUi), inicializálja a zenét (musicPlayer), beállítja az életerőcsíkokat (healthbarP1, healthbarP2), meghívja az initPlayers-t, és beállítja a fegyverek spawnolását (weaponLoader.set).
  + initPlayers(): Inicializálja a két játékos karaktert. Beállítja a megfelelő irányítási eszközt (billentyűzet/egér vagy gamepad) a passedData alapján, majd inicializálja a playerController komponenst mindkét játékoson a kiválasztott skinnel, a spawn pozícióval, az életerővel és a kezdő nézési iránnyal.
* **Jutalomkezelés (giveReward())**
  + A meccs végén, a győztes meghatározása után hívódik meg.
  + Véletlenszerűen dönt skin vagy zene jutalom kiosztásáról a lootGiver komponens segítségével, figyelembe véve a pályát és a játékos már meglévő tárgyait.
  + Ha sikeres a jutalom generálása, frissíti a játékos adatait helyben (passedData), és (ha nincs fejlesztői módban) elküldi az információt az API-nak a szerveren való rögzítéshez (APIManager.AddSkinLoot vagy APIManager.AddMusicLoot).

##### Pálya és Környezet

###### Pályabetöltő és Konfiguráló – mapLoader.css

Ez a szkript felelős a kiválasztott harci pálya ("ham", "prac", "jap") dinamikus betöltéséért és beállításáért a harc kezdetekor. Ez magában foglalja a pálya vizuális rétegeinek, ütközőinek (colliders), és a pályaspecifikus fényeknek a konfigurálását, valamint az animált pályaelemek kezelését.

A loadMap() metódus letörli az előző pályát, beállítja a megfelelő spriteokat, beállítja a pályaspecifikus tartozékokat és eltárolja az aktív pálya nevét.

Az unloadMap() metódus letiltja az összes pályaspecifikus Collider-t és nullára állítja az összes pálya réteg sprite-ját a layerUpdate segítségével, biztosítva a tiszta állapotot a következő pálya betöltése előtt.

###### Spawn Pontok adattárolója – spawnPositions.cs

Ez a szkript egy egyszerű adattároló komponens. Fő célja, hogy a Unity Editorban beállítható Vector2 típusú változókban tárolja a különböző pályákhoz tartozó fegyver spawnpointokat.

##### Játékos Alapvető Vezérlése és Állapota

###### Játékos Mozgás és Akció Vezérlő – playerController.cs

Ez a szkript a játékos karakter központi vezérlője. Fogadja az InputAction.CallbackContext-eket (amiket a Unity Input System küld a PlayerInput komponensen keresztül), és ezek alapján irányítja a karakter mozgását, ugrását, célzását, fegyverhasználatát és egyéb alapvető akcióit. Kezeli a különböző irányítási sémákat (billentyűzet+egér vs. gamepad) és a karakter vizuális megjelenítésének alapjait (sprite tükrözése).

* Inicializálás (init): A gameManager hívja meg a harc/kör kezdetekor. Beállítja a játékos kezdőpozícióját, életerejét (meghívja a playerHealth.init-et), skinjét (a bodyAnim szkripten keresztül), a gamepad használatának jelzőjét (gamepad), a gamepad holtjátékát (deadzone), és lekéri a fegyvertartó (gunHolder) referenciáját.
* Mozgás (Move, FixedUpdate): A Move metódus fogadja a mozgás inputot (vízszintes és függőleges tengely). Az értékeket eltárolja (horizontal, vertical). A FixedUpdate-ben a horizontal érték és a speed alapján beállítja a Rigidbody2D vízszintes sebességét (rb.velocity), megvalósítva ezzel a karakter oldalirányú mozgását. A moving változó jelzi az animáció számára, hogy a karakter éppen mozog-e.
* Ugrás (Jump): A Jump metódus fogadja az ugrás inputot. Ha a játékos a földön van (groundCheck.getGrounded()), akkor beállítja a Rigidbody2D függőleges sebességét a jumpingPower értékre, elindítva az ugrást. (Megjegyzés: A kód tartalmaz coyoteTime és jumpBuffer logikára utaló változókat, de azok teljes implementációja nem látható a Jump metódusban, a FixedUpdate kezeli a coyoteCount csökkentését).
* Célzás (Look, Update): A Look (csak gamepad) és az Update (egér esetén) metódusok beállítják az aimDirection vektort, ami a célzás irányát mutatja. Gamepad esetén figyelembe veszi a holtjátékot (deadzoneCheck). Az Update minden képkockában meghívja a gunHolder.lookAt() metódust, hogy a fegyver a célzás irányába nézzen.
* Sprite Tükrözés (flipSprite, moveDirection): Az aimDirection vízszintes komponense alapján (aimDirection.x > 0) a flipSprite metódus megfordítja (flipX = true/false) a karakter testének (body) sprite-ját, hogy mindig a célzás irányába nézzen. A moveDirection segédfüggvény beállítja a forwardMotion logikai változót, ami jelzi, hogy a karakter a nézési irányával megegyező vagy ellentétes irányba mozog-e (animációhoz lehet hasznos).
* Fegyverkezelés (Fire, AltFire, Equip, Drop): Ezek a metódusok fogadják a fegyverhasználathoz kapcsolódó inputokat. A Fire és AltFire beállít egy shooting / altShooting logikai változót (amit a FixedUpdate használ a folyamatos tüzeléshez) és közvetlenül is meghívja a gunHolder megfelelő tüzelési metódusát, ha az input performed állapotba kerül (pl. gombnyomás). Az Equip és Drop a gunHolder fegyver felvétel/letétel funkcióit hívják meg.
* Létrahasználat (Climb, OnTriggerEnter2D, OnTriggerExit2D): Az OnTrigger... metódusok beállítják a laddered logikai változót, ha a játékos létrával érintkezik. A Climb metódus (amit az Update hív) lehetővé teszi a függőleges mozgást a létrán, ha a játékos a felfelé irányt nyomja (vertical > 0) és éppen létrán van.
* Halálzóna (OnTriggerEnter2D): Ha a játékos "death" taggel ellátott triggerbe lép, meghívja a playerHealth.death() metódust.

Felhasználói Dokumentáció

## Rendszerkövetelmények

# Webes alkalmazás

Ahhoz, hogy a Bullet Hell weboldalát zökkenőmentesen használhassa, az alábbi minimális rendszerkövetelményeknek kell megfelelnie az Ön eszközének és szoftvereinek:

1. **Internetkapcsolat:** Stabil internetkapcsolat (vezetékes vagy vezeték nélküli)
2. **Támogatott eszközök**

* Asztali számítógép
* Laptop
* Táblagép
* Okostelefon

1. **Támogatott operációs rendszerek**

* Windows
* MacOS
* Linux (általánosan használt disztribúció)
* iOS
* Android

1. **Webböngészők**: Általános felhasználású webböngésző

# Asztali alkalmazás

1. **Internetkapcsolat**: Stabil internetkapcsolat (vezetékes vagy vezeték nélküli)
2. **Támogatott eszközök:**

* Asztali számítógép
* Laptop

1. **Támogatott operációs rendszerek:**

* Windows

## Telepítési útmutató

# Webes alkalmazás

Előfeltételek

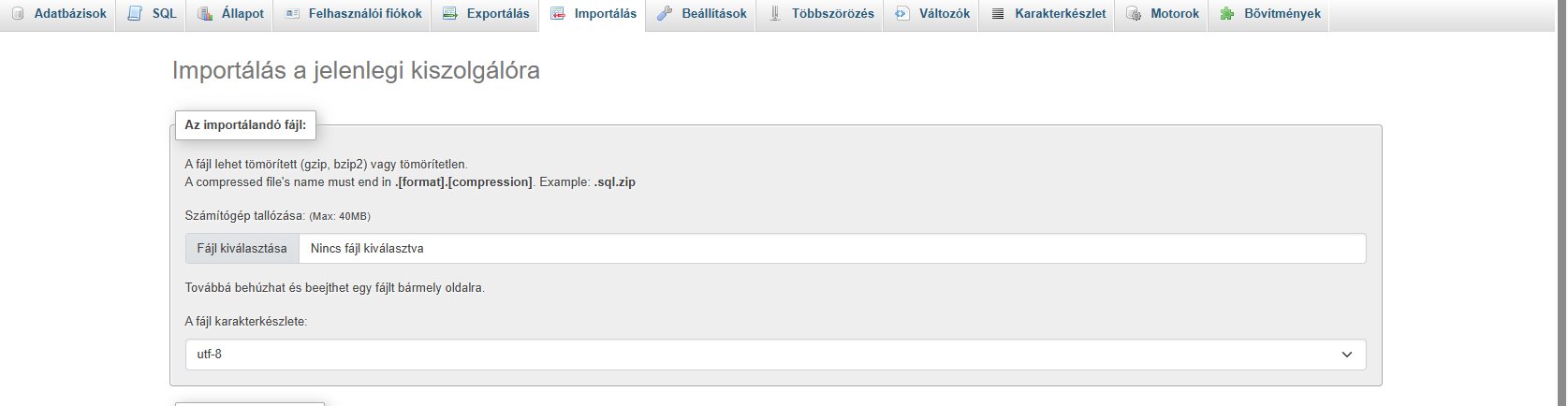
* Helyi webszerver: A weboldalt üzemeltető rendszeren futnia kell egy modern webszervernek PHP-val (Apache ajánlott), és a MariaDB adatbázis-kezelőnek. Az útmutató a XAMPP használatát javasolja az egyszerűség kedvéért, de más módszerrel is feltelepíthetőek és használhatóak a szükséges szoftverek. Győződjön meg róla, hogy a webszerver és a MariaDB szolgáltatások sikeresen elindultak.
* Projekt Fájlok: Szüksége lesz a Bullet Hell projekt /web mappájára, illetve /db mappában található bullet\_hell.sql fájlra.

Telepítési lépések

A webszerver és MariaDB adatbázis kezelő elindítása és konfigurálása után importálja a fent említett adatbázisfájlt (source bullet\_hell.sql;). XAMPP használata esetén navigáljon a phpMyAdmin felületre. Ez általában a http://localhost/phpmyadmin címen érhető el.



Kattintson az "Importálás" (Import) fülre a felső menüsorban.



Az "Importálandó fájl" (File to import) szekcióban kattintson a "Fájl kiválasztása" (Choose File / Browse) gombra. Navigáljon a projekt mappáján belül a /db könyvtárba, és válassza ki a bullet\_hell.sql fájlt.

Hagyja a többi beállítást az alapértelmezett értéken (a formátum SQL legyen).

**Weboldal Fájlok Másolása:**

Keresse meg a webszerver gyökérkönyvtárát (pl. C:\xampp\htdocs, /var/www/html/ vagy /srv/http/).

Másolja be a Bullet Hell projekt weboldalának teljes mappáját a webszerver gyökérkönvtárába, vagy a weboldalnak létrehozott DocumentRootmappába.

**Weboldal elérése**

Nyissa meg a webböngészőjét

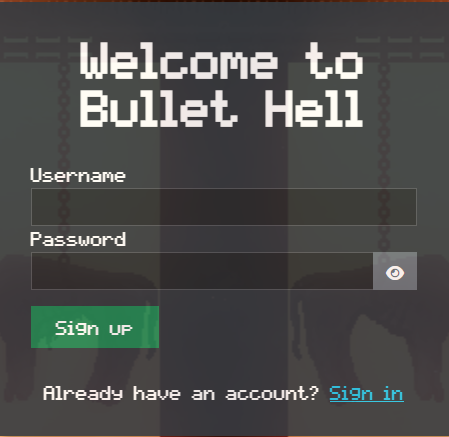
Írja be a címsorba a http:/*localhost/bullet\_hel/web* címet, és a weboldal látható lesz.

Weboldal használata

Ez a fejezet bemutatja a Bullet Hell weboldal főbb funkcióinak használatát. Segít eligazodni a különböző menüpontok között és megérteni az egyes funkciók működését.

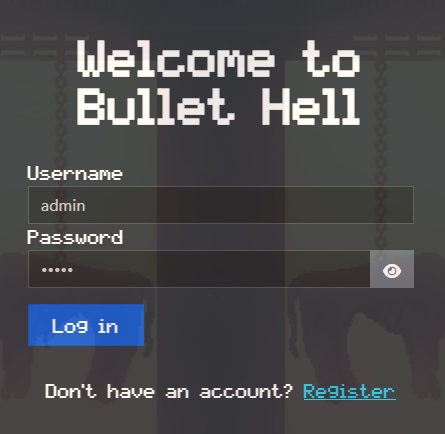
Regisztrációs űrlap

#### Regisztráció

Ahhoz, hogy menthesse az eredményeit a játékban, nyomon követhesse a statisztikáit, és hozzáférjen a weboldal összes funkciójához (mint a rangsor vagy a saját profil), létre kell hozzon egy felhasználói fiókot. A regisztrációhoz csak egy felhasználónév és jelszó kell.

A kitöltéshez írjon be egy egyedi felhasználónevet a ’Username’ mezőbe. Ezután adjon meg egy biztonságos jelszót. Ha mindent helyesen adott meg, a Sign up gomb megnyomása átirányítja a főoldalra.

#### Bejelentkezés

Miután sikeresen regisztrált, a létrehozott felhasználónevével és jelszavával tud bejelentkezni a weboldalra és a játékba. A bejelentkezés szükséges a személyes statisztikái, a rangsorban elfoglalt helyezése és a megszerzett kinézetek megtekintéséhez. A kitöltéshez írja be a korábban létrehozott egyedi felhasználónevét a ’Username’ mezőbe, majd az ehhez létrehozott jelszavát is. A Log in gombra kattintáskor az oldal átirányítja a főoldalra.

#### Saját profil oldal

Bejelentkezés után a jobb felső sarokban a felhasználónevére kattintva jut el erre az oldalra.Ez az Ön személyes központja a weboldalon belül, ahol megtekintheti a játékban elért eredményeit és kezelheti fiókja egyes beállításait.

Képernyőkép Leírás: A képen a Profil oldal látható. A bal oldali sávban a felhasználó karaktere és neve látható. A jobb oldali, nagyobb területen felül a játékos statisztikái ("Stats") jelennek meg több sorban, alatta pedig a jelszóváltoztatási ("Change password") lehetőség található űrlappal.

##### Statisztikák

Ez a szekció összefoglalja az Ön játékbeli teljesítményét. Az itt látható adatok automatikusan frissülnek a játékban elért eredményei alapján.

* Points (Pontszám): Az Ön által a játékban összegyűjtött összes pont.
* Winrate (Győzelmi Arány): Megmutatja, hogy a lejátszott mérkőzéseinek hány százalékát nyerte meg.
* Games Played (Lejátszott Játékok): Az Ön által eddig lejátszott összes mérkőzés száma.
* Total Kills (Összes Ölés): Az Ön által a játékban elért összes kiiktatás (kill) száma.
* Total Deaths (Összes Halál): Az Ön összes halálozásának száma a játékban.
* Aktuálisan Használt Zene (A kép jobb alsó sarkában látható "CRT\_HEAD1"): Itt láthatja annak a zenei csomagnak (music pack) a nevét, amelyet Ön a játékban kiválasztott, és amely a menükben vagy a játék egyes részein hallható.

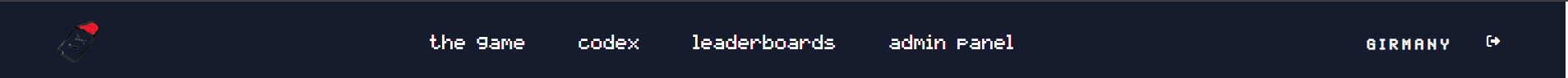
##### Jelszó módosítása

Itt lehetősége van megváltoztatni a fiókjához tartozó jelszót. A biztonság érdekében a rendszer kéri a jelenlegi jelszavát is.

* Current Password (Jelenlegi jelszó): Ebbe a mezőbe írja be a jelenleg érvényes jelszavát.
* New Password (Új jelszó): Ide írja be az új jelszót, amelyet használni szeretne.
* Repeat New Password (Új jelszó megerősítése): Írja be újra pontosan ugyanazt az új jelszót, hogy elkerülje az elgépelést.

#### Navigációs sáv (fejléc)

A weboldal tetején található navigációs sáv (vagy fejléc) biztosítja a könnyű és gyors navigációt a különböző fő részek között. Ez a sáv általában minden oldalon látható, így bármikor könnyedén válthat a funkciók között.

Képernyőkép Leírás: A képen a weboldal sötét hátterű fejléce látható. Bal oldalon egy logó, középen a menüpontok ("the game", "codex", "leaderboards", "admin panel"), jobb oldalon pedig a bejelentkezett felhasználó neve ("GIRMANY") és egy kijelentkezés ikon látható.

1. Logó (Bal oldal):
   1. Itt található a Bullet Hell játék logója.
   2. Funkció: A logóra kattintva visszajuthat a weboldal kezdőlapjára.
2. Főmenü (Középen):
   1. Itt találhatók a weboldal legfontosabb szekcióinak linkjei.
   2. "the game": A játékról szóló általános leírást, bemutatót tartalmazó oldalra vezet.
   3. "codex": A játék enciklopédiája. Ide kattintva érheti el a pályák, fegyverek, zenék, karakterkinézetek és mechanikák részletes leírásait. (Ennek részletes használatát később ismertetjük.)
   4. "leaderboards": A rangsor oldalra vezet, ahol megtekintheti a játékosok rangsorolt listáját pontszám alapján.
   5. "admin panel": Ez a menüpont csak akkor látható, ha Ön adminisztrátori jogosultsággal rendelkezik. Erre kattintva érheti el az adminisztrációs felületet, ahol a játék és a weboldal adatait kezelheti. (Erről külön leírás szól az adminisztrátoroknak.)
3. Felhasználói Rész (Jobb oldal):
   1. Felhasználónév (pl. "GIRMANY"): Ha be van jelentkezve, itt láthatja a saját felhasználónevét.
      1. Funkció: A felhasználónevére kattintva közvetlenül a saját Profil oldalára navigálhat.
   2. Kijelentkezés Gomb (Ikon): A felhasználóneve mellett található egy ikon
      1. Funkció: Erre az ikonra kattintva tud kijelentkezni a fiókjából. Kijelentkezés után visszakerül a bejelentkezési oldalra.

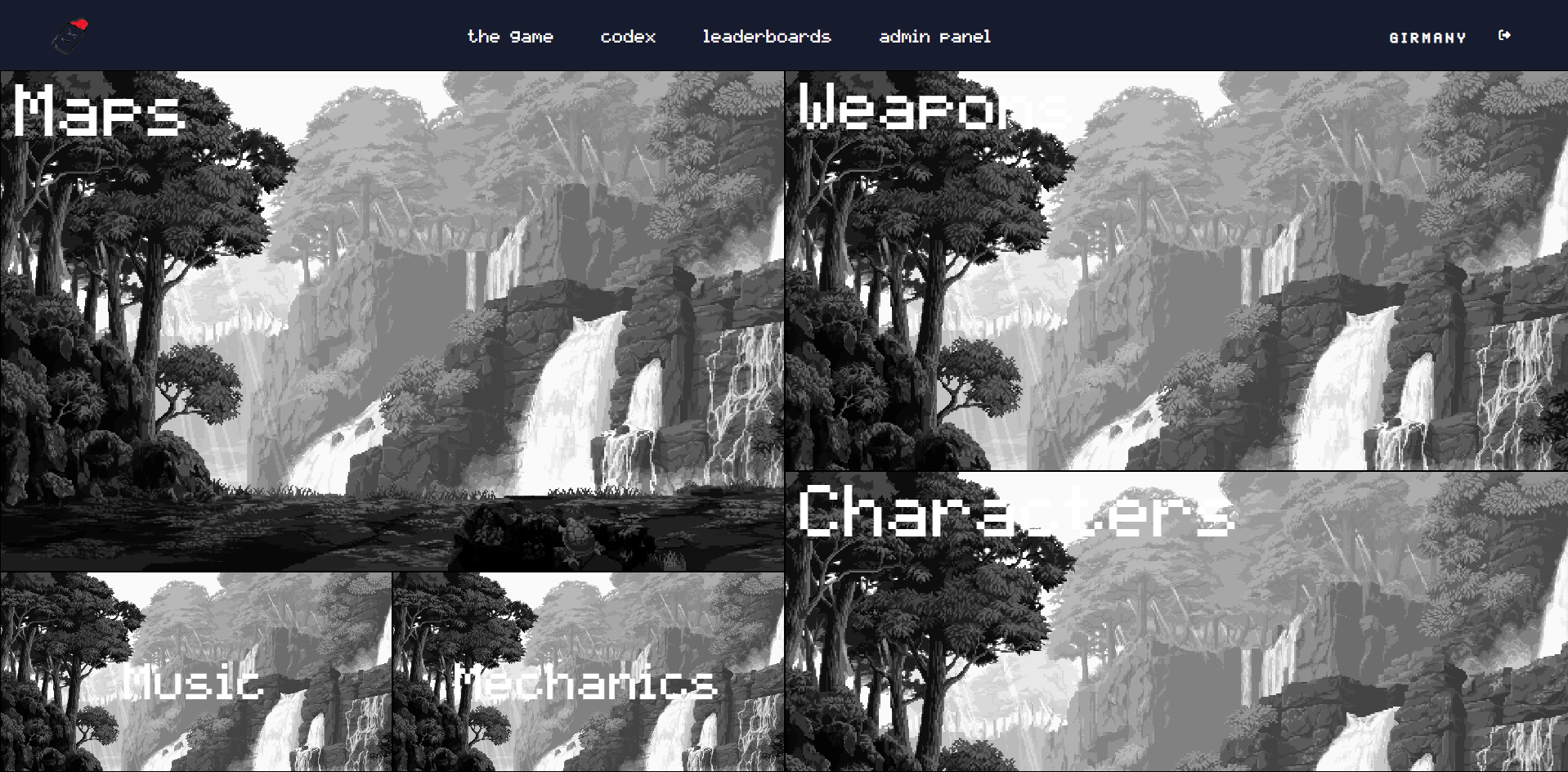
#### Rangsor

A Rangsor oldal a Bullet Hell felhasználóinak rangsorát mutatja be. Itt láthatja, hogyan állnak a játékosok egymáshoz képest a játékban elért pontszámuk és győzelmi arányuk alapján, és hol helyezkedik el Ön a többi játékos között. Az oldal eléréséhez kattintson a navigációs sávban található „leaderboards” menüpontra.

Képernyőkép Leírás: A képen a Rangsor oldal látható. Felül egy cím ("LEADERBOARD") és egy legördülő menü ("Number of players per page") található. Alatta egy táblázat sorolja a játékosokat (Rank, Username, Points, Winrate, Games played, Kills, Deaths oszlopokkal). Az első három sor és a bejelentkezett játékos sora ki van emelve. Alul lapozó nyilak és oldalszám jelző látható.

* A Táblázat Oszlopai:
  + Rank: A játékos helyezése a rangsorban.
  + Username: A játékos felhasználóneve.
  + Points: A játékos által elért összes pontszám. A rangsor alapértelmezetten ez alapján van rendezve, csökkenő sorrendben.
  + Winrate: A játékos győzelmi aránya százalékban.
  + Games played: A játékos által lejátszott összes mérkőzés száma.
  + Kills: A játékos által elért összes kiiktatás (ölés).
  + Deaths: A játékos összes halálozásának száma.
* Kiemelések:
  + Top 3 Játékos: A rangsor első három helyezettje vizuálisan ki van emelve (a képen eltérő háttérszínnel), hogy könnyen azonosíthatók legyenek a legjobbak.
  + Saját Helyezés: Ha Ön szerepel az aktuálisan megjelenített listán, az Ön sora egy zöld kerettel van kiemelve, így gyorsan megtalálhatja saját magát a rangsorban.
* Navigáció és Beállítások:
  + "Number of players per page" (Játékosok száma oldalanként): A táblázat felett található legördülő menü segítségével beállíthatja, hogy egyszerre hány játékos adatait szeretné látni egy oldalon (a képen ez 15-re van állítva).
* Lapozás: Mivel a rangsorban sok játékos lehet, az adatok több oldalra vannak tördelve. A táblázat alján található:
  + Balra Nyíl (<): Visszalapozhat az előző oldalra.
  + Jobbra Nyíl (>): Előre lapozhat a következő oldalra.
  + Oldalszám Jelző (pl. "1/34"): Megmutatja, hogy Ön éppen hányadik oldalt látja a rangsorból, és összesen hány oldal van.

#### Kódex

A Kódex a Bullet Hell játék világának és elemeinek részletes enciklopédiája. Itt mindent megtudhat a pályákról, fegyverekről, karakterekről, zenékről és a játék egyéb fontos mechanikáiról. Amikor a Kódex menüpontra kattint, egy áttekintő felület fogadja Önt, ahol a játék különböző kategóriái látványos, csempeszerű elrendezésben jelennek meg. Minden csempe egy-egy fő témakört képvisel. Képernyőkép Leírás: A képen a Kódex főoldala látható, ahol öt nagy csempe jeleníti meg a fő kategóriákat (Maps, Weapons, Characters, Music, Mechanics) egy háttérképen. A csempék alapértelmezetten enyhén áttetszőek vagy szürkések.

A főoldalon a következő kategóriák közül választhat:

* Maps (Pályák): Információk és képek a játék különböző harctereiről.
* Weapons (Fegyverek): Részletes adatok a játékban elérhető összes fegyverről (sebzés, tűzgyorsaság stb.).
* Characters (Karakterek/Kinézetek): A választható karakterkinézetek (skinek) bemutatása.
* Music (Zenék): A játékban felcsendülő zenei csomagok listája és leírása.
* Mechanics (Mechanikák): A játékmenet alapvető és különlegesebb működési elveinek magyarázata.

A következő alfejezetekben részletesen bemutatjuk, mit talál az egyes kategóriákon.

#### Maps (pályák)

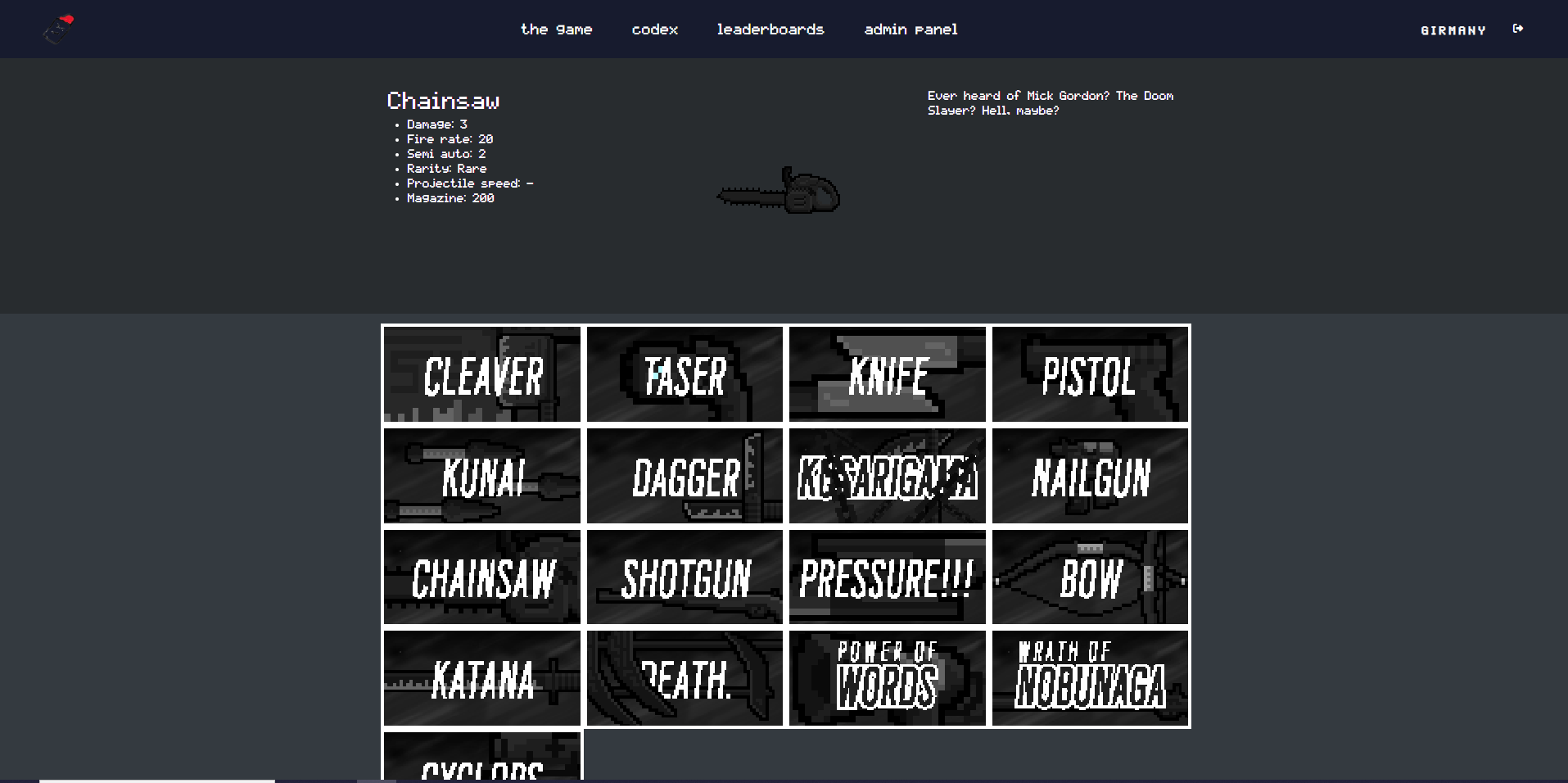
Miután a Kódex főoldalán a "Maps" csempére kattintott, a játék pályáit bemutató oldalra érkezik. Itt vizuálisan megismerkedhet a különböző harcterekkel és elolvashatja azok rövid leírását.

Az oldal teljes részét egy képnézegető foglalja el, amely teljes képernyős nézetben mutatja be a játék pályáit egyenként. Mindig csak egy pálya látható egyszerre. Képernyőkép Leírás: A képen a "Palace of the Setting Sun" nevű pálya látható teljes nézetben. A kép bal és jobb szélén navigációs nyilak (<, >) találhatók. A kép alatt, a jobb alsó sarokban olvasható a pálya neve és egy rövid leírás.

A különböző pályák közötti váltáshoz használja a képnézegető bal és jobb oldalán található fehér nyilakat: Balra Nyíl (<): Erre kattintva az előző pályára léphet a listában. Jobbra Nyíl (>): Erre kattintva a következő pályára léphet a listában. A nyilak segítségével végig lapozhatja az összes elérhető pályát.

#### Fegyverek

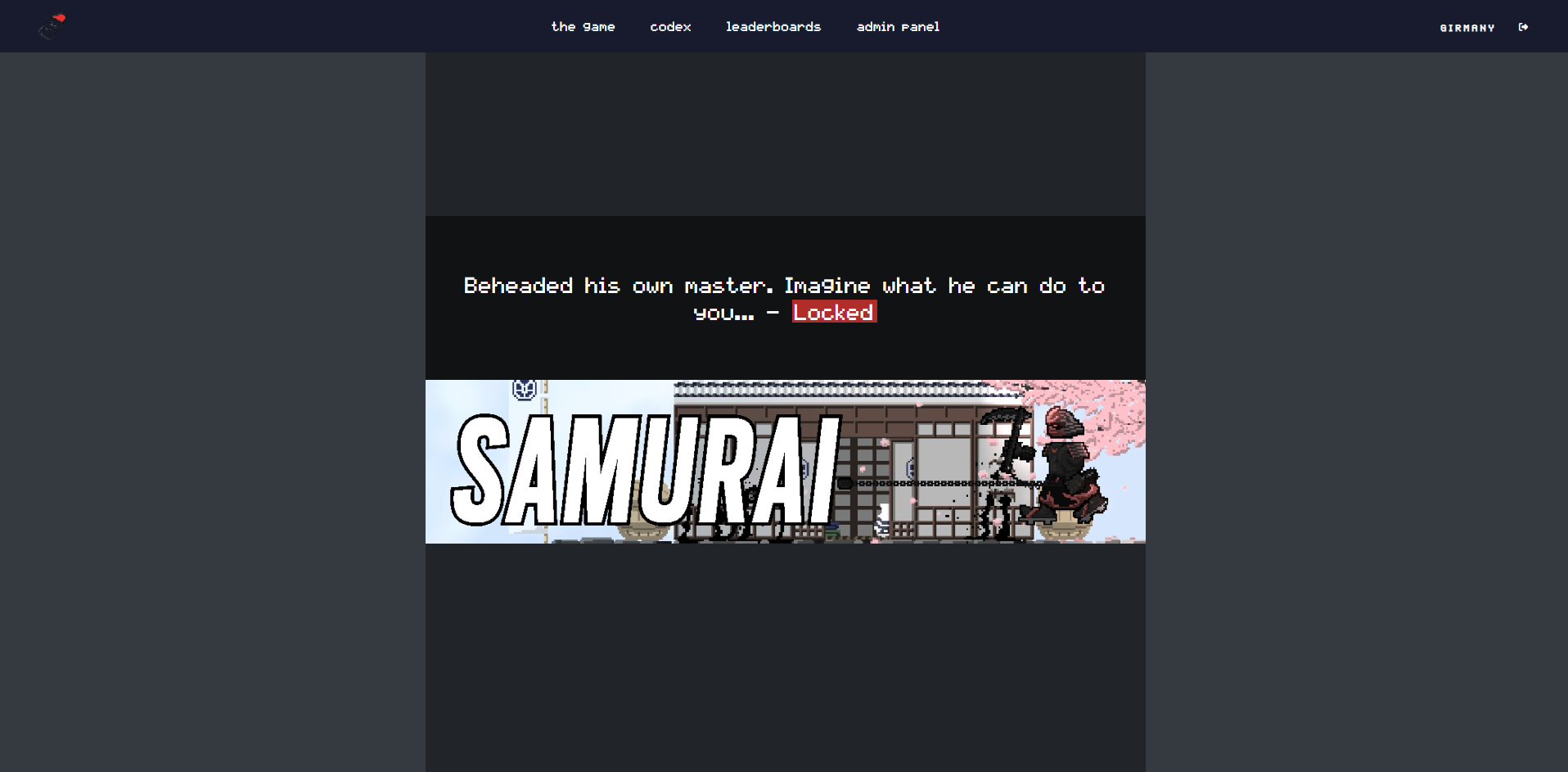
Ha a Kódex főoldalán a "Weapons" csempére kattintott, a játék fegyverarzenálját bemutató oldalra jut. Itt részletes információkat találhat minden egyes fegyverről, beleértve azok statisztikáit és leírását. Az oldal két fő részre tagolódik: felső rész a kiemelt fegyver. Ez a nagyobb felső terület mindig az aktuálisan kiválasztott fegyver részletes adatait mutatja.

Az alsó rész a fegyverválasztó. Ez a rész egy rácsos (csempés) elrendezésben listázza az összes elérhető fegyvert, lehetővé téve a böngészést és a kiválasztást. Képernyőkép Leírás: A képen a Fegyverek kódex oldala látható. Felül, a "Kiemelt Fegyver" részben bal oldalon a "Chainsaw" nevű fegyver statisztikái (Damage, Fire rate, stb.) láthatók listázva, középen a fegyver kis képe (ikonja), jobb oldalon pedig a fegyver leírása olvasható. Az oldal alsó részén egy rácsban több fegyver neve látható csempéken.

Amikor kiválaszt egy fegyvert az alsó listából, annak adatai a kiemelt részben jelennek meg.

* Fegyver Neve: Az aktuálisan kiválasztott fegyver neve (pl. "Cleaver").
* Statisztikák (Stats): Részletes lista a fegyver játékbeli tulajdonságairól:
  + Damage: A fegyver által okozott sebzés mértéke.
  + Fire rate: A fegyver tűzgyorsasága (milyen sűrűn tud lőni).
  + Semi auto: Jelzi, hogy a fegyver félautomata-e (1 = igen, 0 = nem).
  + Rarity: A fegyver ritkasági szintje
  + Projectile speed: A fegyver lövedékének sebessége (ha van lövedéke).
  + Magazine: A fegyver tárkapacitása
* Fegyver Képe/Ikonja: A fegyver vizuális megjelenítése a játékban.
* Leírás: Rövid szöveges leírás a fegyverről, annak használatáról vagy történetéről.

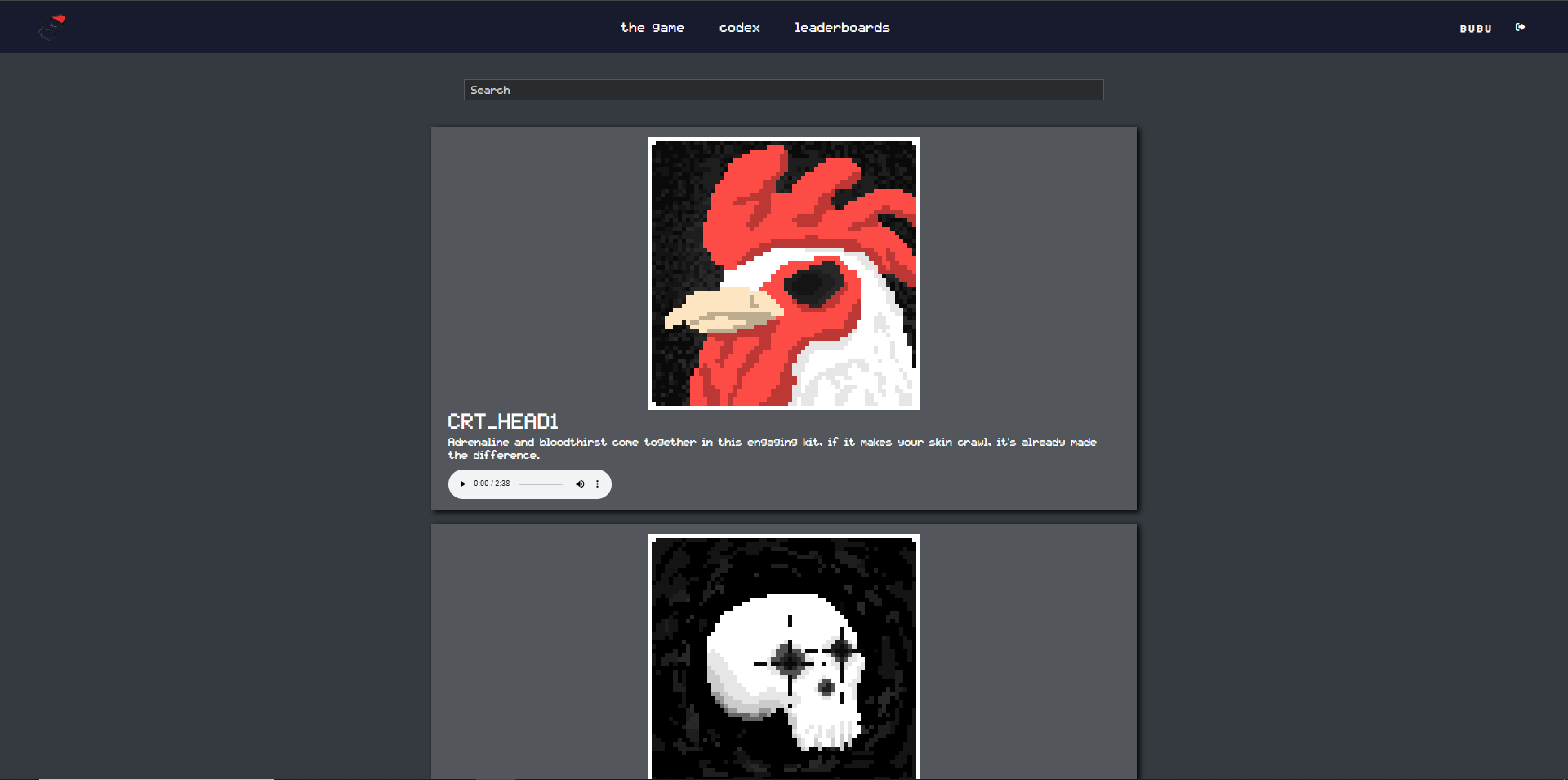
#### Karakterek (kinézetek)

A Kódex főoldalán a „Characters” csempére kattintva érheti el ezt a szekciót, amely a játékban használható különböző karakterkinézeteket (skinek) mutatja be. Itt áttekintheti az elérhető választékot és ellenőrizheti, melyeket birtokolja már. A játékban elérhető karakterkinézetek itt vízszintes sávokban jelennek meg, egymás alatt sorakozva. Minden sáv egy adott kinézetet reprezentál, és tartalmazza a karakter képét, és hátterét. Képernyőkép Leírás: A képen a Karakterek/Kinézetek oldal látható. Vízszintes sávokban jelennek meg az egyes kinézetek.

A különböző kinézetek részleteinek megtekintéséhez húzza az egérmutatót a karakter sávja fölé. Az ekkor beúszó szöveg tartalmazza a karakter leírását és egy állapotjelzőt, hogy az adott karakter ön által birtokolva van-e (Locked vagy Unlocked).

#### Zenék

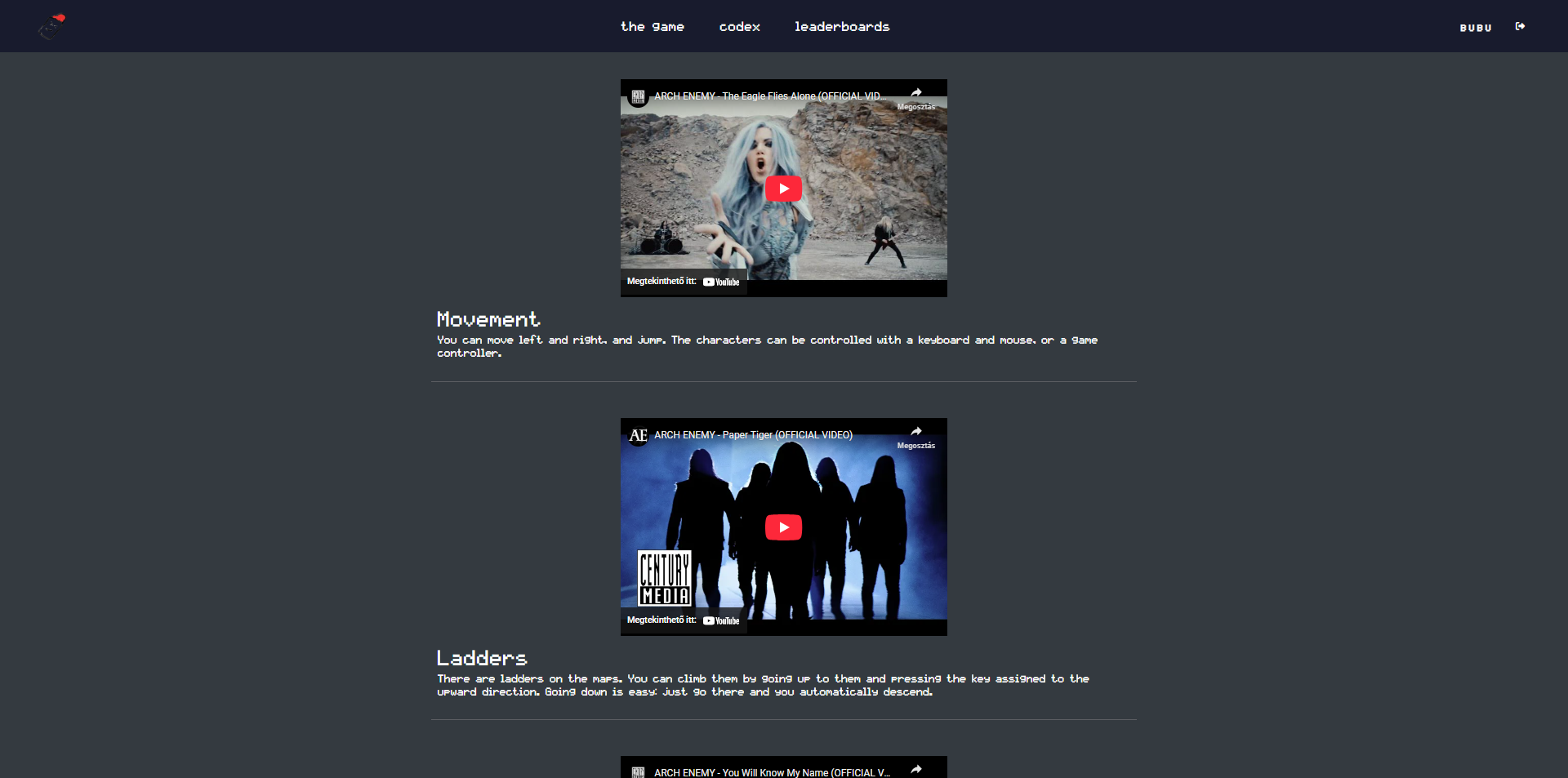
A Kódex főoldalán a „Music” csempére kattintva juthat el a zenéket bemutató oldalra. A zenék egymás alatt csempékben sorakozva. A csempék fő eleme a zene borítóképe, alatta a cím és egy rövid leírás a zenéről. Lehetősége van belehallgatni a zenei csomag zenéjébe a leírás alatt elhelyezett zenelejátszóval.

Képernyőkép leírás: A képen a zenei csempék láthatóak egy zenelejátszóval és felül egy keresőmezővel.

Keresőmező használata:

A keresőmezőbe kezdje el gépelni a keresett zenei csomag nevét, és ha van találat, a zenei csomag feltűnik:

#### Mechanikák

A Kódex "Mechanics" szekciója segít megérteni a Bullet Hell játék alapvető működési elveit és játékmenetbeli elemeit. Ha a Kódex főoldalán a "Mechanics" csempére kattintott, erre az oldalra érkezik, ahol részletes magyarázatokat és vizuális bemutatókat talál a mozgásról, harcról és egyéb fontos funkciókról. Képernyőkép Leírás: A képen a Játékmechanikák oldal egy részlete látható szekció. Felül egy videólejátszó látható, alatta pedig a mozgásról szóló angol nyelvű szöveges magyarázat.

#### Admin panel

Az Admin Panel egy különleges felület a weboldalon belül, amely **kizárólag adminisztrátori jogosultsággal rendelkező felhasználók számára érhető el**. Ez a felület biztosítja az eszközöket a játék és a weboldal adatainak kezeléséhez és karbantartásához. Ez a funkció átlagos játék felhasználóknak nem érhető el.

* Belépés
  1. Amennyiben Ön adminisztrátori jogosultsággal rendelkezik és be van jelentkezve, keresse meg a felső navigációs sávban az "admin panel" menüpontot.
  2. A menüpontra kattintás után nem közvetlenül az adminisztrációs eszközökhöz jut, hanem egy **biztonsági ellenőrző oldalra** kerül. A képen szöveg, képernyőkép, Betűtípus, szoftver látható

     Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.Ezen az oldalon a rendszer **ismételten kéri az Ön adminisztrátori jelszavát** egy egyszerű űrlapon keresztül. Ez egy további biztonsági lépés annak érdekében, hogy csak az arra jogosult személy férhessen hozzá az adminisztrációs funkciókhoz, még akkor is, ha a fiókja esetleg bejelentkezve maradt egy számítógépen.
  3. Ha a helyes jelszót adta meg, az oldal továbbirányítja az admin felületre.
* A képen szöveg, képernyőkép, tervezés látható

  Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.**Felhasználók felület**
  1. **Bal Oldali Navigáció**: Itt találhatók az adminisztrációs menüpontok (User management, Weapon management, stb.), valamint egy link ("Back to homepage") a weboldal főoldalára való visszatéréshez.
  2. **Fő Tartalmi Terület:** Itt jelenik meg az aktuálisan kiválasztott kezelési feladat, ebben az esetben a felhasználók listája.
  3. **"Number of users per page":** A táblázat feletti legördülő menü segítségével beállíthatja, hogy egy oldalon hány felhasználó adatai jelenjenek meg egyszerre
  4. **Lapozás**: A táblázat alatt lapozó gombok találhatók (hasonlóan a Rangsorhoz), amelyekkel a felhasználók listájának további oldalaira léphet, ha több felhasználó van, mint az oldalanként beállított szám.
  5. Az oszlopfejlécekre kattintva rendezheti a listát az adott tulajdonság alapján
  6. Egy felhasználó végleges eltávolításához kattintson a felhasználó sorának végén található kuka ikonra (Delete oszlop).
* **A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

  Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.Fegyverkezelés felület**
  1. A táblázat lehetővé teszi az összes fegyver és azok részletes statisztikáinak (ID, Név, Fájlnév, Sebzés, Tűzgyorsaság, Félautomata, Ritkaság, Lövedéksebesség, Tárkapacitás, Leírás, Fejlesztői infó) gyors áttekintését.
  2. Egy fegyver végleges eltávolításához kattintson a fegyver sorának végén található kuka ikonra (Delete oszlop).
  3. A jobb oldali "Add New Weapon" űrlap segítségével új fegyvereket adhat hozzá a játékhoz.
     + Miután kitöltötte az összes szükséges mezőt, kattintson az **"Add Weapon"** gombra. A rendszer feldolgozza az adatokat és hozzáadja az új fegyvert az adatbázishoz, amely ezt követően megjelenik a bal oldali táblázatban is. Sikeres hozzáadásról vagy esetleges hibákról a rendszer visszajelzést ad.
* **Pályakezelő felület**
  1. A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

     Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.A Pályakezelő felület az Admin Panel része, amelyet a bal oldali navigációs menüben a **"Map management"** opcióra kattintva érhet el. Ezen az oldalon kezelheti a játékban elérhető pályákat: áttekintheti a meglévőket, újakat adhat hozzá, vagy törölheti a már nem szükségeseket. A felület egy listázó táblázatból és egy hozzáadási űrlapból áll.
  2. A felület bal/középső részén található "Maps" táblázatban láthatja az összes, jelenleg az adatbázisban szereplő pályát.
  3. Ha egy pályát el szeretne távolítani a játékból, keresse meg a törölni kívánt pályát a táblázatban. Kattintson a pálya sorának végén, a Delete oszlopban található kuka ikonra.
  4. Új pálya hozzáadásához használja a jobb oldalon található "Add New Map" űrlapot. Töltse ki a kitöltendő mezőket az új pálya adatival. Miután kitöltötte az adatokat, kattintson az **"Add Map"** gombra. A rendszer hozzáadja az új pályát az adatbázishoz, és az megjelenik a bal oldali listában is.
* A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

  Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.**Zenei csomag kezelőfelület**
  1. Az Admin Panel ezen szekciója lehetővé teszi Önnek, mint adminisztrátornak, a játékban elérhető zenei csomagok kezelését. Itt áttekintheti a meglévő csomagokat, törölheti azokat, vagy újakat adhat hozzá a rendszerhez.
  2. Új zenei csomag hozzáadásához használja a jobb oldalon található "Add New Music Pack" űrlapot. Töltse ki a kitöltendő mezőket az új zenei csomag adatival. Miután kitöltötte az adatokat, kattintson az **"Add Music Pack"** gombra. A rendszer hozzáadja az új zenét az adatbázishoz, és az megjelenik a bal oldali listában is.
  3. Ha egy zenei csomagot el szeretne távolítani a játékból, keresse meg a törölni kívánt zenét a táblázatban. Kattintson a zene sorának végén, a Delete oszlopban található kuka ikon.
* A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

  Előfordulhat, hogy a mesterséges intelligencia által létrehozott tartalom helytelen.**Karakterek/kinézetek kezelése**
  1. Ez a szekció az adminisztrátorok számára biztosít felületet a játékban elérhető karakterkinézetek (skinek) kezeléséhez. Itt áttekintheti a meglévő kinézeteket, törölheti azokat, és újakat adhat hozzá a játékhoz.
  2. Új karakter hozzáadásához használja a jobb oldalon található "Add New Character" űrlapot. Töltse ki a kitöltendő mezőket az új karakter adatival. Miután kitöltötte az adatokat, kattintson az **"Add Character"** gombra. A rendszer hozzáadja az új karaktert az adatbázishoz, és az megjelenik a bal oldali listában is.
  3. Ha egy karaktert el szeretne távolítani a játékból, keresse meg a törölni kívánt karaktert a táblázatban. Kattintson a karakter sorának végén, a Delete oszlopban található kuka ikonra.

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