**Web technológiák 2.**

Féléves feladat

*Videójáték nyilvántartó rendszer*

Készítette: **Gerőcs Gergő**

Neptun kód: **FEU2E5**

**Tartalomjegyzék**

[2 Felhasználói felület 3](#_Toc199844108)

[2.1 Üdvözlés 3](#_Toc199844109)

[2.2 Bejelentkezés 3](#_Toc199844110)

[2.3 Adminisztrátori menü 3](#_Toc199844111)

[2.3.1 Ügyintézők Listázása 3](#_Toc199844112)

[2.3.2 Ügyintézők Létrehozása 3](#_Toc199844113)

[2.3.3 Ügyintézők Módosítása 3](#_Toc199844114)

[2.3.4 Adminisztrátorok Listázása 3](#_Toc199844115)

[2.3.5 Adminisztrátorok Létrehozása 3](#_Toc199844116)

[2.3.6 Adminisztrátorok Módosítása 3](#_Toc199844117)

[2.3.7 Adminisztrátorok Törlése 3](#_Toc199844118)

[2.4 Ügyintézői menü 3](#_Toc199844119)

[2.4.1 Játékok Listázása 3](#_Toc199844120)

[2.4.2 Játékok Keresése 3](#_Toc199844121)

[2.4.3 Játékok Létrehozása 3](#_Toc199844122)

[2.4.4 Játékok Módosítása 3](#_Toc199844123)

[2.4.5 Játékok Törlése 3](#_Toc199844124)

[3 Kód 3](#_Toc199844125)

[3.1 Model 3](#_Toc199844126)

[3.2 Entity 3](#_Toc199844127)

[3.3 Controller 3](#_Toc199844128)

[3.4 Component 3](#_Toc199844129)

[3.5 Service 4](#_Toc199844130)

# Felhasználói felület

## Üdvözlés

A GameHouse egy online videójáték nyilvántartó rendszer, melyet ügyintézők és adminisztrátorok használnak.

Az ügyintézők játékokat tudnak hozzáadni, módosítani, listázni és törölni, illetve különböző paraméterek alapján tudnak keresni.

Az adminisztrátorok az ügyintézők adatait tudják szerkeszteni, illetve új ügyintézőt hozzá adni, más adminisztrátorok adatait szerkeszteni, csak a superAdmin tudja.

Az alkalmazás elindításakor a felhasználó az üdvözlő menübe kerül, innen kiválaszthatja, hogy ügyintézői vagy adminisztrátori bejelentkezést.

Az adott gombra kattintva a felhasználó az adott bejelentkező felületre kerül.

## Bejelentkezés

Az ügyintézőknek, valamint az adminisztrátoroknak saját bejelentkezési felülete van.

Az ügyintézőknek jelszóra és felhasználónévre van szükségük a belépéshez, ezeket nem saját maguk, hanem az adminisztrátorok adják meg.

Az adminisztrátoroknak jelszóra, felhasználónévre és egy kódra van szükségük.

## Adminisztrátori menü

Ha az adminisztrátor superAdmin-ként lép be, akkor elérhetővé válik az adminisztrátorok szerkesztése, létre lehet hozni új adminisztrátor fiókot, módosítani, listázni, illetve törölni, a már létezőt.

SuperAdmin fiókot nem lehet létrehozni, automatikusan jön létre indításkor, továbbá nem lehet módosítani és törölni sem.

A képen szöveg, képernyőkép, Betűtípus, Téglalap látható

Előfordulhat, hogy az AI által létrehozott tartalom helytelen.Ha az adott illető, sima adminisztrátorként lép be, akkor számára csak az ügyintézőkkel kapcsolatos menük érhetőek el.

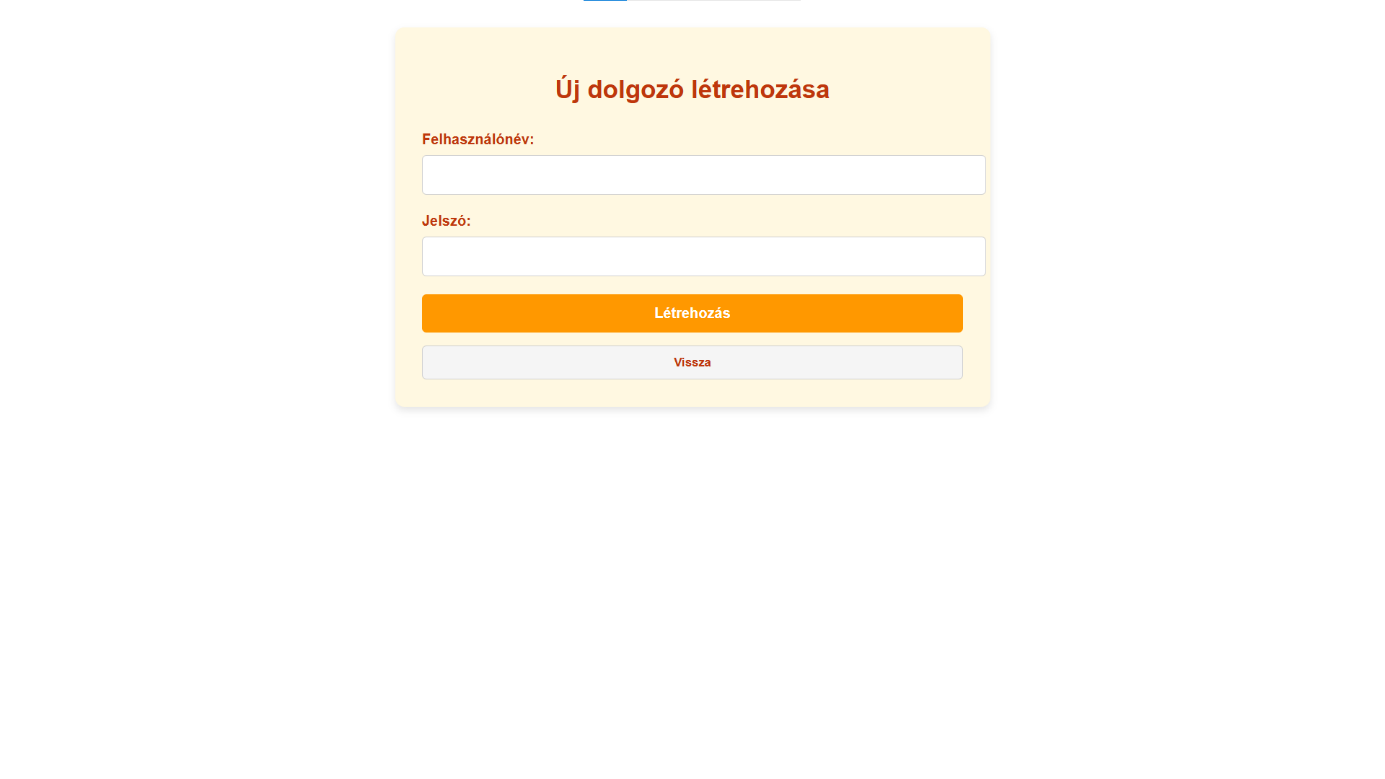
### Ügyintézők Listázása

Ebben a menüben az adminisztrátorok, láthatják felsorolva az ügyintézőket, és azok adatait.



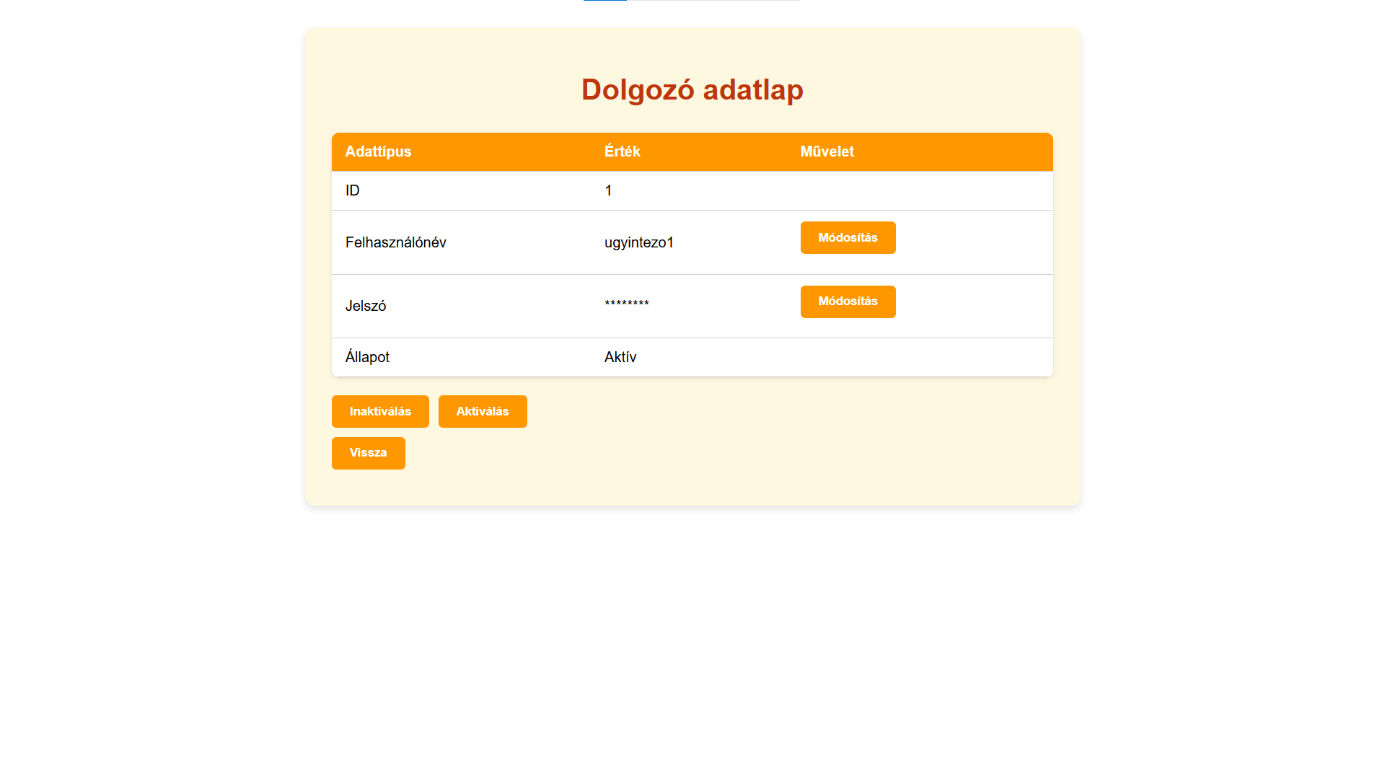
### Ügyintézők Létrehozása

Ezen menüpontban az adminisztrátoroknak lehetőségük van, új ügyintézői fiók létrehozására.



### Ügyintézők Módosítása

A módosítás felületen lehetséges az ügyintézők adatait módosítani, például kaphatnak új jelszót vagy felhasználónevet. A módosítás Id alapján történik.

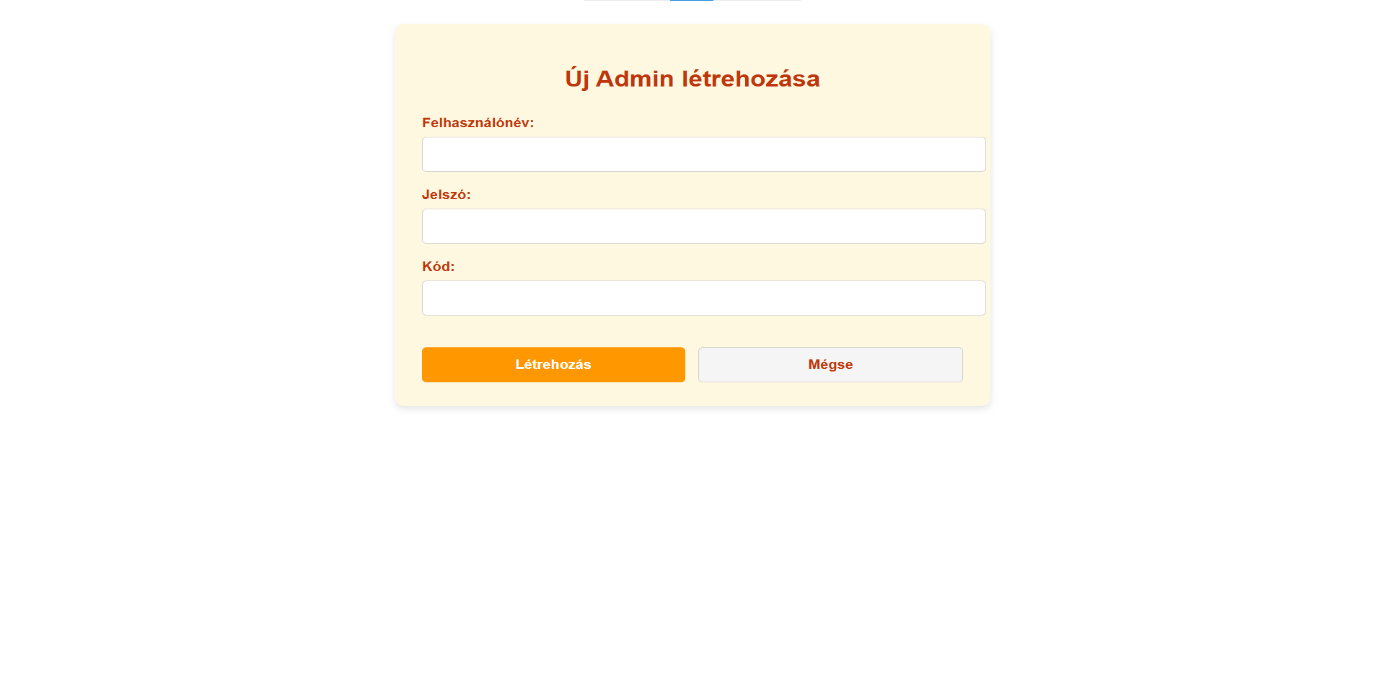


### Adminisztrátorok Listázása

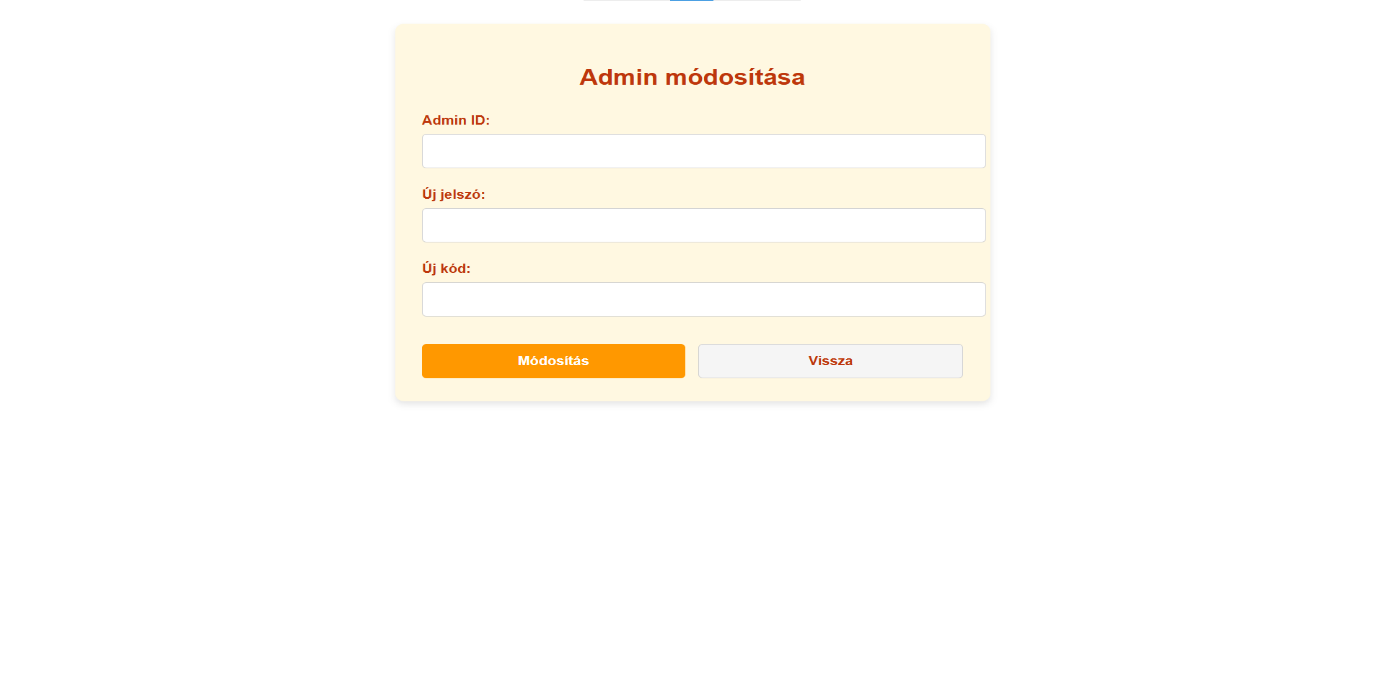
Ezen felületen a jogosultsággal rendelkező felhasználók megtekinthetik az adminisztrátorok listáját, mely tartalmazza a felhasználónevet, Id-t, illetve a státuszt, ami lehet aktív vagy inaktív.

### Adminisztrátorok Létrehozása

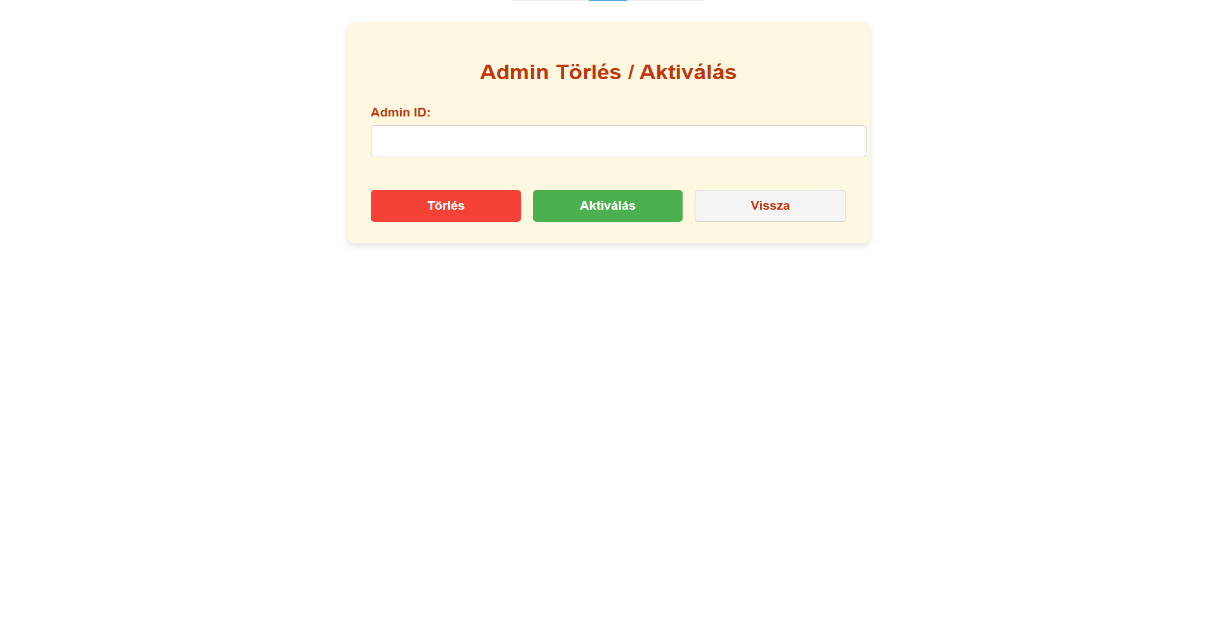
SuperAdminként létrehozható új adminisztrátori fiók, ehhez szükség van a következőkre:

* Felhasználónév
* Kód
* jelszó

### Adminisztrátorok Módosítása

Ezen felület a superAdmin képes az adminisztrátorok adatainak a módosítására, Id alapján.

### Adminisztrátorok Törlése

Az adminisztrátori fiókok törlésére, szintén a superAdmin képes, ez a törlés, csak státusz váltással jár.

## Ügyintézői menü

Az ügyintézői menüben lehetséges a játékokkal kapcsolatos műveletek végrehajtása. Ezen műveletek:

* Listázás
* Keresés
* Létrehozás
* Módosítás
* Törlés



### Játékok Listázása

Ezen a felületen láthatóak a játékok kilistázva adataikkal együtt.

Ezek a következők:

* Sorszám
* Cím
* Hozzáadás dátuma
* Kategória
* Platform

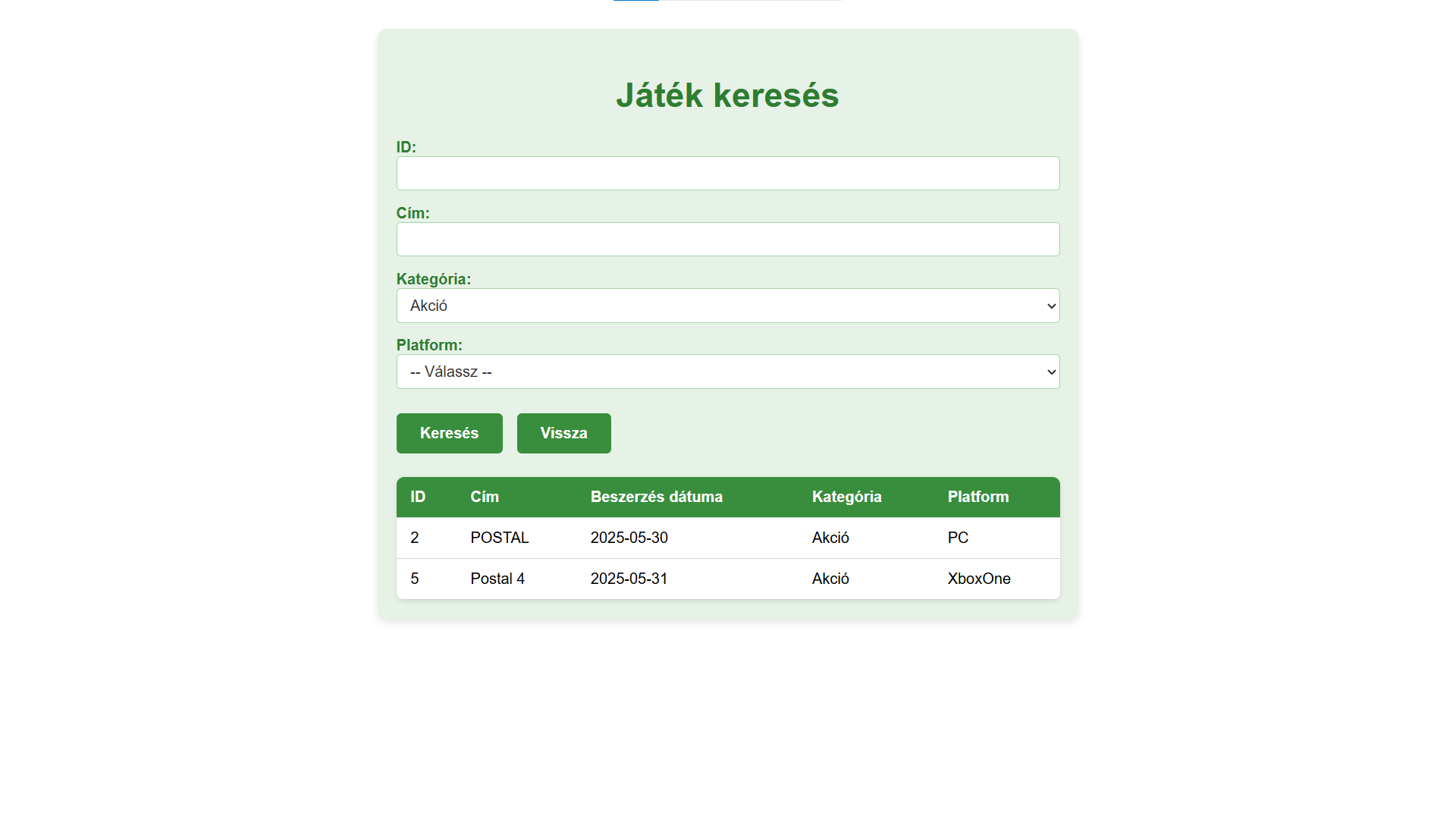
A platform és kategória adatot az ügyintéző egy előre megadott listából választhatja ki.

A képen szöveg, képernyőkép, Betűtípus, szoftver látható

Előfordulhat, hogy az AI által létrehozott tartalom helytelen.

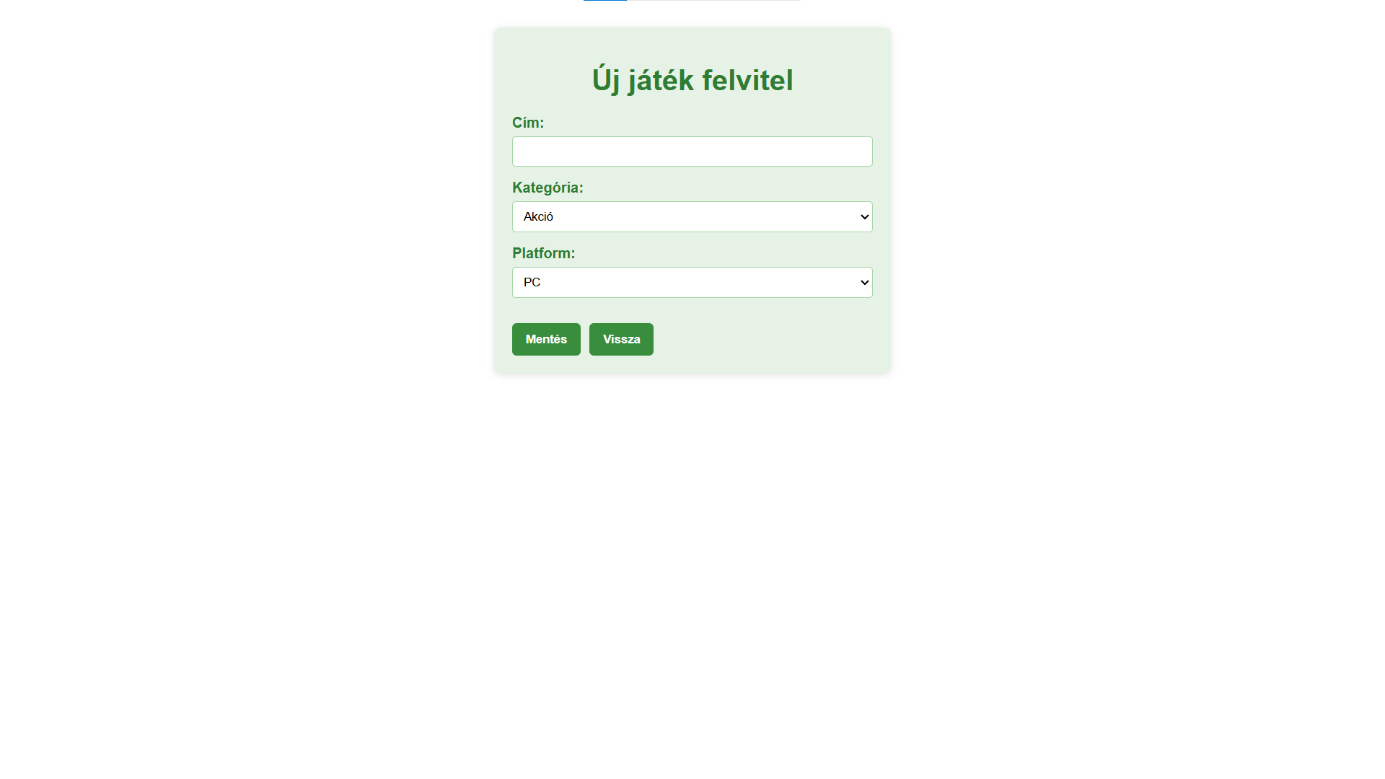
### Játékok Keresése

A játékok keresése többféleképpen is történhet, lehet Id, Cím, Kategória és Platform alapján keresni.

Fontos megjegyezni, hogy egymás felett helyezkednek el a keresési feltételek mezői, és ez a fentről lefelé haladó sorrend maga a fontossági sorrend.

### Játékok Létrehozása

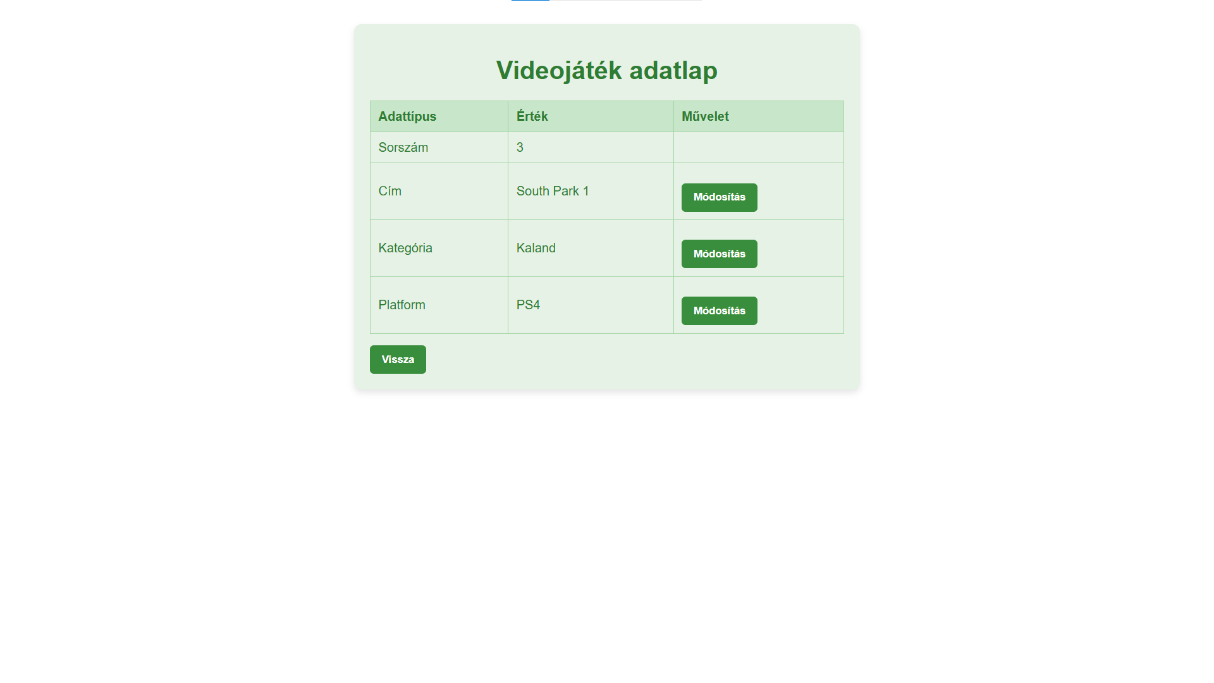
Az ügyintéző hozzáadhat új játékokat, ebben az esetben meg kell adni a játék címét, kategóriáját, illetve a platformot amin megjelent.

A hozzáadás dátuma automatikusan kerül hozzáadásra, illetve a sorszámot az adatbázis generálja.

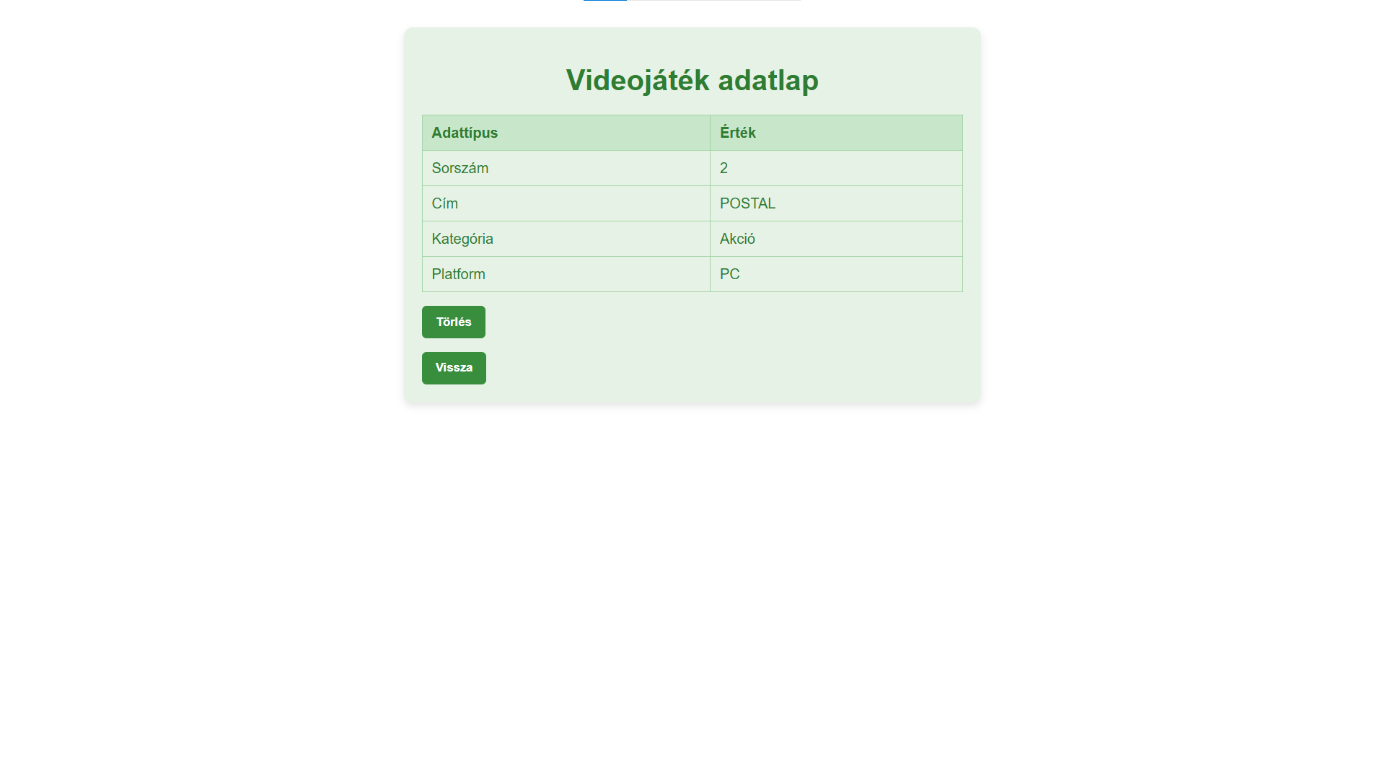
### Játékok Módosítása

Ebben a menüben lehet a játékok adatait módosítani azonosító alapján. Ezen adatokat egyesével lehet változtatni.

Módosítható adatok:

* Cím
* Kategória (az előre meghatározott kategóriák közül lehet választani)
* Platform (az előre meghatározott platformok közül lehet választani)

### Játékok Törlése

A játékok törlését az ügyintéző végzi és sorszám alapján történik.

# Kód

A GameHouse alkalmazás frontendje Angular keretrendszerrel, TypeScript nyelven készült. A weboldalak egységes és reszponzív megjelenését saját CSS szabályok biztosítják.

A backend TypeScriptben, Node.js környezetben, Express keretrendszerrel, TypeORM leképezéssel és MySQL adatbázissal készült. Az alkalmazás támogatja a session alapú hitelesítést és bcrypt algoritmussal kezeli a jelszavakat.

## Model

A projekt gyökerében létrehozott *models* jegyzék index.ts fájlja tartalmazza a kliens-szerveradatátvitelkor használt DTO-kat.

Kód:

export interface VersionDTO {

    version: string;

}

export interface GameDTO {

  sorszam: number;

  cim: string;

  beszerzes\_datuma: Date;

  kategoria: string;

  platform: string;

}

export interface StaffDTO {

id: number;

username: string;

password: string;

isActive: boolean;

}

export interface StaffResponseDTO {

  id: number;

  username: string;

  isActive: boolean;

}

export interface AdminDTO {

  id: number;

  username: string;

  password: string;

  code: string;

  isActive: boolean;

}

export interface AdminResponseDTO {

  id: number;

  username: string;

  isActive: boolean;

}

## Entity

A feladat elkészítése során 3 entitást használtam:

* Admin: adminisztrátor, tulajdonságai:
  + id, username, password, code, isActive
* Game: videójáték, tulajdonságai:
  + sorszam, cím, beszerzes\_datuma, kategoria, platform
* Staff
  + id, username, password, isActive

Az isActive minden esetben egy igaz/hamis érték, amelyet törléshez használtam, mivel a törlés ebben az esetben, csak státusz váltással jár.

A játékokhoz tartozó kategória és platform mezők csak előre meghatározott értékeket vehetnek fel, mivel ezek enum típusokként kerültek definiálásra a backend oldalon. Így biztosítható, hogy csak a meghatározott értékkészletből lehessen választani, ezzel is növelve az adatintegritást és elkerülve a hibás adatbeviteleket.

**Entity-k:**

Admin.ts entity

@Entity("admin")

export class Admin implements AdminDTO{

    @PrimaryGeneratedColumn()

    id: number;

    @Column()

    username: string;

    @Column()

    password: string;

    @Column()

    code: string;

    @Column({ default: true })

    isActive: boolean;

}

Game.ts entity

@Entity()

export class Game implements GameDTO{

    @PrimaryGeneratedColumn()

    sorszam: number;

    @Column()

    cim: string;

    @Column({ type: 'date', nullable: false })

    beszerzes\_datuma: Date;

    @Column({

        type: 'enum',

        enum: JatekKategoria

      })

      kategoria: JatekKategoria;

    @Column({

        type: 'enum',

        enum: Platform

      })

      platform: Platform;

}

Staff.ts entity

@Entity()

export class Staff implements StaffDTO{

    @PrimaryGeneratedColumn()

    id: number;

    @Column()

    username: string;

    @Column()

    password: string;

    @Column()

    isActive: boolean;

}

**Enumok:**

platform.enum.ts

export enum Platform {

    PC = 'PC',

    PS5 = 'PS5',

    PS3 = 'PS3',

    PS4 = 'PS4',

    PS2 = 'PS2',

    XBOX360 = 'Xbox360',

    XBOXOne = 'XboxOne',

    SWITCH = 'Switch'

  }

game-category.enum.ts

export enum JatekKategoria {

    AKCIO = 'Akció',

    KALAND = 'Kaland',

    RPG = 'RPG',

    STRATEGIA = 'Stratégia',

    SPORT = 'Sport',

    VERSENY = 'Verseny',

    HORROR = 'Horror',

    PUZZLE = 'Puzzle',

    SZIMULATOR = 'Szimulátor',

    MMO = 'MMO',

    PARTY = 'Party',

    ZENEI = 'Zenei',

    INDIE = 'Indie',

    SCIFI = 'Sci-fi',

    FANTASY = 'Fantasy'

  }

## Controller

## Component

A frontend legfontosabb elemeit a következő Angular komponensek alkotják:

Néhány példa:

## Service