

SZOFTVERFEJLESZTŐ ÉS -TESZTELŐ

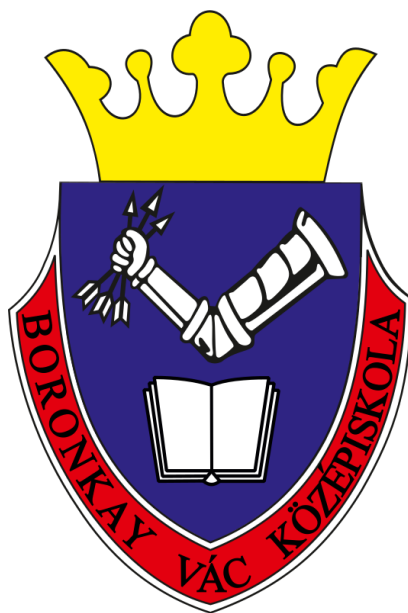
SZAKDOLGOZAT

ALBERT BENCE

RABI RÉKA

2025

VÁCI SZAKKÉPZÉSI CENTRUM
BORONKAY GYÖRGY
MŰSZAKI TECHNIKUM ÉS GIMNÁZIUM



SZAKDOLGOZAT

Capygames Store

Konzulens: Gyombolainé Cserny Zsuzsa

Készítette: Albert Bence

Rabi Réka

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.

.....

Albert Bence

.....

Rabi Réka

Konzultációs lap

Vizsgálók neve: Albert Bence, Rabi Réka

Szakdolgozat címe: Capygames Store

Program által nyújtott szolgáltatások:

- Játékok böngészése, vásárlása
- Bejelentkezés, regisztráció
- Játékok rendszerigényeinek megtekintése
- Birtokolt játékok nyilvántartása
- Felhasználói profil megtekintése, szerkesztése

Sorszám	A konzultáció időpontja	A konzulens aláírása
1.	2024. október 11.	
2.	2024. november 15.	
3.	2024. december 13.	
4.	2025. január 17.	
5.	2025. február 14.	
6.	2025. március 14.	

A szakdolgozat beadható:

Vác, 2025.

.....

Konzulens

A szakdolgozatot átvettem:

Vác, 2025.

.....

A szakképzést folytató
intézmény felelőse

Tartalomjegyzék

Hallgatói nyilatkozat	2
Konzultációs lap	3
Tartalomjegyzék.....	4
1 Témaválasztás	8
2 Feladatspecifikáció.....	9
2.1 Eredeti feladatspecifikáció.....	9
2.2 Módosított feladatspecifikáció	9
3 Munkamegosztás	10
4 Fejlesztői dokumentáció	11
4.1 Fejlesztői környezet	11
4.1.1 Visual Studio	11
4.1.2 phpMyAdmin	11
4.1.3 Unity.....	11
4.1.4 Notepad++	11
4.1.5 Visual Studio Code	11
4.2 Használt programok, programozási nyelvek, technológiák.....	12
4.2.1 C#	12
4.2.2 HTML.....	12
4.2.3 CSS.....	12
4.2.4 JavaScript	12
4.2.5 .NET Standard 2.1	12
4.2.6 MariaDB	12
4.2.7 CapyScript	13
4.2.8 Odin Inspector	13

4.2.9 vFolders2 és vHierarchy2	13
4.2.10 XAMPP	13
4.2.11 PHP.....	13
4.3 Adatbázis.....	14
4.3.1 Az adatbázis alapadatai	14
4.3.2 Az adatbázis szerkezete	14
4.3.3 Az adatbázis táblái	15
4.4 Szerepkörök	25
4.4.1 Vendég felhasználó	25
4.4.2 Felhasználó	25
4.5 Alkalmazások szerkezete	26
4.5.1 Asztali alkalmazás	26
4.5.2 Weboldal.....	30
4.5.3 Rest API.....	33
4.6 Funkciók bemutatása	34
4.6.1 Játéklista	34
4.6.2 Kívánságlista.....	34
4.6.3 Könyvtár	34
4.6.4 Profil.....	34
4.6.5 Játék oldal	34
4.6.6 Kosár	34
4.7 Kapcsolati diagram.....	35
4.7.1 Asztali alkalmazás	35
4.7.2 Web.....	37
4.7.3 Rest API	37
4.8 Bemeneti értékek ellenőrzése	38

4.8.1 Hitelesítési adatok ellenőrzése	38
4.8.2 JWT token ellenőrzés	38
4.8.3 Kosár és rendelés műveletek bemeneti ellenőrzése	38
4.8.4 Termékek lekérdezése	39
4.8.5 Általános hibakezelés.....	39
4.9 Biztonsági intézkedések.....	39
4.9.1 JWT alapú hitelesítés (JSON Web Token)	39
4.9.2 Jelszóhash-elés (bcryptjs)	40
4.9.3 Adatvalidálás a bemeneti oldalon	40
4.9.4 Middleware védelem.....	40
4.9.5 HTTPS használat ajánlott	40
4.9.6 CORS védelem.....	40
4.9.7 Hibakezelés és információ visszatartás	41
4.9.8 Session kezelés.....	41
4.9.9 Időtartam alapú token lejárat.....	41
5 Felhasználói dokumentáció	42
5.1 Rendszerkövetelmények.....	42
5.1.1 Asztali alkalmazás rendszerkövetelményei	42
5.2 Telepítési útmutató	42
5.2.1 Asztali alkalmazás telepítése	42
5.2.2 Szerverkörnyezet telepítése	45
5.3 Felületek bemutatása	46
5.3.1 Asztali alkalmazás felületei	46
5.3.2 Weblap felületei.....	51
6 Konklúziók	55
7 Irodalomjegyzék.....	56

7.1 Online források	56
7.2 Online eszközök	56
8 Mellékletek	56

1 Témaválasztás

A projekt ihletét a Steam webáruház adta. A projektünk, a Capygames Store a Steam webáruház alternatívája lenne. De miért kéne bárkinek is a Capygames Store a Steam webáruház helyett, ha már a Steam rég elterjedt és piacvezető? Nos, a Steamen nehéz indie-ként érvényesülni, így a Capygames Store kifejezetten az indie fejlesztőknek nyújtana egy vonzó alternatívát. A Steamen egy játék regisztrálása 100€-ba kerül, és ezen túl a Steam a bevétel 30%-át is megtartja. A Capygames Store azzal próbálja meg segíteni a feltörekvő játékfejlesztőket, hogy a regisztrációs díjat 10€-ra csökkenti, valamint csak a bevétel 20%-át tartja meg. Továbbá a Steamhez hasonlóan, a játékot érintő adók továbbra is a webáruház feléből kerülnek levonásra, így a fejlesztő a bruttó bevétel 80%-át kapja. A regisztrációs díj pusztán a spamelés kivédése érdekében marad meg.

A Capygames Store név egy belsős poénból ered, miszerint „A capybara a legmenőbb állat a világon”. Ebből a poénból alakult ki a CapySomething formátum, amit az évek során több különböző helyen használtunk, így mivel egy játék webáruházat, avagy angolul game store-t akartunk csinálni, egyből felmerült a Capygames Store név, és végül megmaradtunk ennél.

2 Feladatspecifikáció

2.1 Eredeti feladatspecifikáció

Az eredeti feladatspecifikáció szerint a projekt tartalmaz egy asztali, web, és mobil alkalmazást, melyek közül mindegyik lehetőséget ad regisztrációra, játékok böngészésére, azok vásárlására, felhasználók profiljának megtekintésére, értékelésre, kívánságlisták és könyvtárak tárolására. Ezeken felül a felhasználók feltölthetnek profilképet. A játékok szintén rendelkeznek képekkel, melyek segítik a felhasználókat választásban. Az asztali alkalmazás ezen túl képes lett volna telepíteni és futtatni a játékok fájlait. A specifikáció ezen verziójában a részalkalmazások kizárólag a RestAPI-val kommunikálnak, egymástól függetlenek.

2.2 Módosított feladatspecifikáció

A fejlesztés megkezdése után nemsokkal az eredeti 3 csoporttag egyike kilépett a csoportból, így módosítottuk a feladatspecifikációt, hogy 2 taggal is megvalósítható legyen. Az eredeti specifikációhoz képes a natív mobil alkalmazást elvetettük, ahogy a képek és játékfájlok tárolását is, valamint az alkalmazások futtatását, mivel ezek jelentették a legnagyobb technikai kihívást. Ezek megvalósítása jelentős mennyiségű kutatást vett volna igénybe, valamint olyan technológiák és megoldások használatát, amikkel még sohasem dolgoztunk. Bár 3 taggal lett volna időn új megoldások megtanulására a projekt kedvéért, ezt 2 taggal már nem mertük megkockáztatni. Később nem várt csúszások miatt úgy határoztunk, hogy az asztali alkalmazáson csak böngészni lehet a játékokat, megvásárolni nem. Így a vásárláshoz bekerült egy átirányítás a weblapra.

3 Munkamegosztás

A projekt során a következő munkamegosztást alkalmaztuk:

- Adatbázis megtervezése és kivitelezése: Albert Bence
- RestAPI megtervezése és kivitelezése: Rabi Réka
- Asztali alkalmazás megtervezése és kivitelezése: Albert Bence
- Asztali alkalmazás tesztelése: Rabi Réka
- Weboldal megtervezése és kivitelezése: Rabi Réka
- Weboldal tesztelése: Albert Bence
- Szerverkörnyezet megtervezése és felállítása: Albert Bence
- Egyedi grafikák készítése: Albert Bence

A tesztelési feladatokat szándékosan úgy osztottuk ki, hogy ne a tervező / kivitelező végezze, hogy a tesztelő elfogulatlanul, friss szemmel állhasson neki a tesztelésnek, és hogy ez még inkább igaz legyen, a teszteléshez feketedobozos rendszer- és elfogadási tesztelést használtunk. A fejlesztési folyamat részeként a fejlesztő (nem a tesztelő) fehérdobozos egység- és integrációs teszteléseket is végzett.

A dokumentáció írását felosztottuk. Mindenki azokat a feladatokat dokumentálja, amelyeken ő dolgozott, így garantálva a dokumentáció pontosságát. A dokumentáció mindkettőnket érintő részét Albert Bence írta.

Ahol csak lehetséges figyelembe vettük minden tag egyéni képességeit, erősségeit, és ezek alapján osztottuk fel a feladatokat.

4 Fejlesztői dokumentáció

4.1 Fejlesztői környezet

4.1.1 Visual Studio

Az asztali alkalmazás fejlesztéséhez szükséges C#-kódok a Visual Studio 2022-ben készültek. A Visual Studio egy, a Windows operációs rendszerre készült fejlesztői környezet, mely rengeteg programozási nyelvet támogat, és könnyen módosítható az igények kielégítésére amennyiben szükséges.

4.1.2 phpMyAdmin

Az adatbázis elkészítéséhez a phpMyAdmin-t használtuk, ami egy webinterfész az adatbázis eléréséhez, és könnyebb, átláthatóbb kezeléséhez. A phpMyAdmin sokat segített az adatbázis elkészítésében, mivel sokkal átláthatóbb módon tudtuk ellenőrizni, hogy az adatbázis a megfelelő módon készült el.

4.1.3 Unity

Az asztali alkalmazás a Unity 6-ban készült. A Unity alapvetően egy videójáték motor, vagyis videójátékok készítésére szánt program, ám remekül alkalmas nem játék alkalmazások készítésére is. Azért választottuk a Unity-t, mert rendkívül könnyen kimondottan személyre szabott UI-t lehet benne készíteni, valamint kimondottan könnyűvé teszi annak a kóddal való összekapcsolását.

4.1.4 Notepad++

A Notepad++ egy szövegszerkesztő, mely el van látva több kódolást segítő funkcióval is, mint például kulcsszavak kiemelésével. A projekt során sokszor használtuk a Notepad++-t, mikor olyan számítógépen kellett kódot szerkeszteni, amelyen más alternatíva nem volt elérhető, vagy amikor csak gyors, apró módosításokat végeztünk, mint például egy elírás kijavítása.

4.1.5 Visual Studio Code

A Visual Studio Code a Visual Studio könnyű változata, kimondottan jó HTML, CSS, JavaScript és PHP támogatással, így a weboldal és a Rest API készítéséhez ezt használtuk.

4.2 Használt programok, programozási nyelvek, technológiák

4.2.1 C#

A C# a Microsoft erősen típusos magas szintű programozási nyelve, melyet az asztali alkalmazás elkészítése során használtunk. Választásunk azért erre esett, mert a nyelv lehetővé teszi, hogy viszonylag könnyen viszonylag bonyolult, hatékony kódot írjunk, valamint a Unity-vel is kompatibilis.

4.2.2 HTML

A weboldal elkészítéséhez a HTML nyelvet használtuk, mely lehetőséget ad rá hogy a weblap elemeit, tartalmát egy jól átlátható, könnyen értelmezhető módon definiáljuk. A HTML alapértelmezetten támogatott a legtöbb böngészőben.

4.2.3 CSS

A weboldalon található HTML kód által generált szerkezet formázásához, díszítéséhez a CSS (Cascading Style Sheets) nyelvet használtuk. A CSS kiváló lehetőséget ad rá, hogy a weboldal kinézetét körülírással határozzuk meg, ahelyett, hogy azt manuálisan le kéne kódolni.

4.2.4 JavaScript

A JavaScript a böngészők által alapértelmezetten támogatott kliens oldali programozási nyelv. A JavaScript-et a weboldalon használtuk, mivel lehetőséget biztosít a HTML és CSS kódok módosítására, így biztosítva, hogy az oldal reagáljon a változásokra.

4.2.5 .NET Standard 2.1

Az asztali alkalmazás készítése során a C# nyelvet a Unity a .NET Standard 2.1-es verziója segítségével értelmezte. A .NET Standard 2.1 választása más .NET verziók helyett biztosítja, hogy az alkalmazás a lehető legstabilabban működjön a lehető legtöbb rendszeren.

4.2.6 MariaDB

Az adatbázis kezeléséhez a MariaDB-t használjuk, mely egy MySQL alapú adatbázis kezelő rendszer. A MariaDB magas fokú kompatibilitással rendelkezik a sztenderd MySQL API-kkal, így ezt választottuk.

4.2.7 CapyScript

A CapyScript egy Albert Bence által fejlesztett és karbantartott Unity Package, mely lényegében előre elkészített kódok összessége. Célja, hogy játékfejlesztés alatt gyakran használt, de alapból nem elérhető funkciókat biztosítson, hogy ne kelljen azokat minden alkalommal leprogramozni. Jelen projekt leginkább a CapyScript beépített Singleton osztályait használta.

4.2.8 Odin Inspector

Az Odin Inspector egy Unity Package, mely a Unity-ban írt kódok számára biztosít lehetőséget, hogy könnyen módosítsák az Inspectorukat. Ezen módosítások általában validálási célt szolgálnak, vagy csak pusztán esztétikailag. Az Odin Inspector kimondottan hasznos a scriptek manuálisan konfigurált bementei értékeinek ellenőrzésére és korlátozására, így csökkentve a bugok kialakulásának esélyét.

4.2.9 vFolders2 és vHierarchy2

A vFolders2 és vHierarchy2 két Unity Package, melyek a Unity Editor kinézetét változtatják meg, valamint hasznos gyorsgombokat adnak hozzá az Editorhoz. A vFolders2 a mappák kezelését dolgozza újra, míg a vHierarchy2 a Hierarchy-ét.

4.2.10 XAMPP

A XAMPP egy előre elkészített szerver csomag, mely tartalmazza többek között a weblap megosztásához szükséges Apache-t, az adatbázis szerverét, valamint a PHP kódok értelmezéséhez szükséges szerveret is, mindezt előre konfigurálva. Míg alapvetően a XAMPP nem ajánlott élő szerverekhez, gyors felállításának köszönhetően tökéletes a teszteléshez valamint a helyi bemutatóhoz.

4.2.11 PHP

A PHP egy szerver oldali programozási nyelv web szolgáltatások számára. Mivel a PHP hatékonyan és egyszerűen képes kommunikálni az adatbázissal, így ezt használtuk mind a weboldal létrehozásakor, mind a RestAPI létrehozásakor.

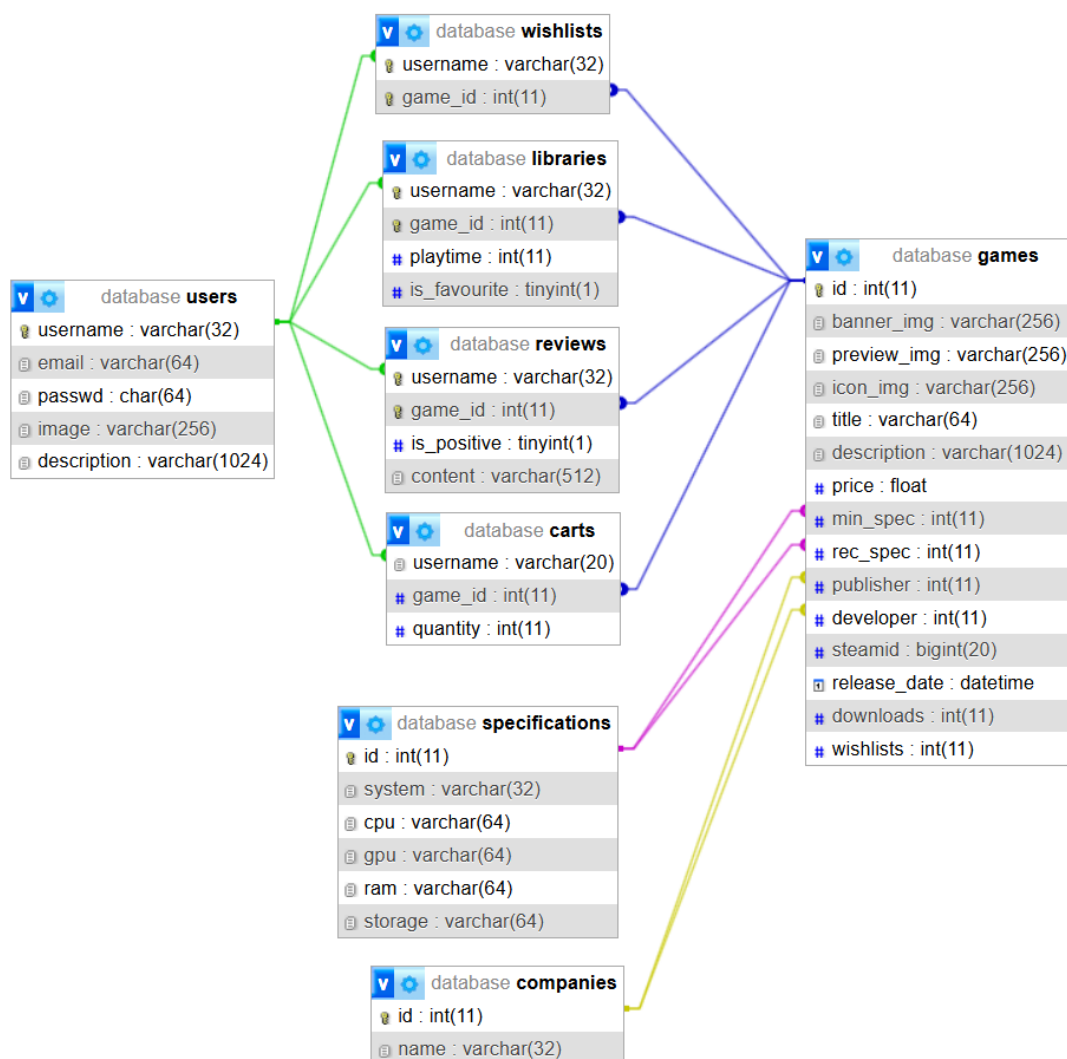
4.3 Adatbázis

4.3.1 Az adatbázis alapadatai

- **Adatbázis szerver:** MariaDB 10.x/11.x vagy kompatibilis MySQL verzió (A fejlesztés során MariaDB 10.4.32-t használtunk)
- **Adatbázis neve:** database
- **Illesztés:** utf8mb4_hungarian_ci
- **Használt adatbázismotor:** InnoDB

```
CREATE DATABASE IF NOT EXISTS `database` DEFAULT CHARACTER  
SET utf8 COLLATE utf8_hungarian_ci;
```

4.3.2 Az adatbázis szerkezete



4.3.3 Az adatbázis táblái

4.3.3.1 Users


A users tábla a felhasználók alapértelmezett adatait tárolja. A passwd mező nem magát a jelszót, hanem a jelszó hash-et tárolja.

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett
1	username 	varchar(32)	utf8_hungarian_ci		Nem	Nincs
2	email	varchar(64)	utf8_hungarian_ci		Nem	Nincs
3	passwd	char(64)	utf8_hungarian_ci		Nem	Nincs
4	image	varchar(256)	utf8_hungarian_ci		Nem	
5	description	varchar(1024)	utf8_hungarian_ci		Nem	

```
CREATE TABLE IF NOT EXISTS `users` (  
  `username` varchar(32) NOT NULL,  
  `email` varchar(64) NOT NULL,  
  `passwd` char(64) NOT NULL,  
  `image` varchar(256) NOT NULL DEFAULT "",  
  `description` varchar(1024) NOT NULL DEFAULT "",  
  PRIMARY KEY (`username`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8  
COLLATE=utf8_hungarian_ci;
```


4.3.3.2 Companies

A companies tábla fejlesztő és kiadó cégeket tartalmaz.

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett
1	id	 int(11)			Nem	Nincs
2	name	varchar(32)	utf8_hungarian_ci		Nem	Nincs

```
CREATE TABLE IF NOT EXISTS `companies` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `name` varchar(32) NOT NULL,  
  PRIMARY KEY (`id`)  
) ENGINE=InnoDB AUTO_INCREMENT DEFAULT CHARSET=utf8  
COLLATE=utf8_hungarian_ci;
```

4.3.3.3 Specifications

A specifications tábla a játékok specifikációit tárolja.

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések
1	id 🔑	int(11)			Nem	<i>Nincs</i>	
2	system	varchar(32)	utf8_hungarian_ci		Nem	Not specified	
3	cpu	varchar(64)	utf8_hungarian_ci		Nem	Not specified	
4	gpu	varchar(64)	utf8_hungarian_ci		Nem	Not specified	
5	ram	varchar(64)	utf8_hungarian_ci		Nem	Not specified	
6	storage	varchar(64)	utf8_hungarian_ci		Nem	Not specified	

```
CREATE TABLE IF NOT EXISTS `specifications` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `system` varchar(32) NOT NULL DEFAULT 'Not specified',  
  `cpu` varchar(64) NOT NULL DEFAULT 'Not specified',  
  `gpu` varchar(64) NOT NULL DEFAULT 'Not specified',  
  `ram` varchar(64) NOT NULL DEFAULT 'Not specified',  
  `storage` varchar(64) NOT NULL DEFAULT 'Not specified',  
  PRIMARY KEY (`id`)  
) ENGINE=InnoDB AUTO_INCREMENT DEFAULT CHARSET=utf8  
COLLATE=utf8_hungarian_ci;
```

4.3.3.4 Games

A games tábla tárolja a webáruház által kínált játékokat.

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett
1	id 🔑	int(11)			Nem	<i>Nincs</i>
2	banner_img	varchar(256)	utf8_hungarian_ci		Nem	
3	preview_img	varchar(256)	utf8_hungarian_ci		Nem	
4	icon_img	varchar(256)	utf8_hungarian_ci		Nem	
5	title	varchar(64)	utf8_hungarian_ci		Nem	Untitled game
6	description	varchar(1024)	utf8_hungarian_ci		Nem	
7	price	float			Nem	0
8	min_spec 🔑	int(11)			Igen	<i>NULL</i>
9	rec_spec 🔑	int(11)			Igen	<i>NULL</i>
10	publisher 🔑	int(11)			Igen	<i>NULL</i>
11	developer 🔑	int(11)			Igen	<i>NULL</i>
12	steamid	bigint(20)			Igen	<i>NULL</i>
13	release_date	datetime			Igen	<i>NULL</i>
14	downloads	int(11)			Nem	0
15	wishlists	int(11)			Nem	0

```



CREATE TABLE IF NOT EXISTS `games` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `banner_img` varchar(256) NOT NULL DEFAULT "",
  `preview_img` varchar(256) NOT NULL DEFAULT "",
  `icon_img` varchar(256) NOT NULL DEFAULT "",
  `title` varchar(64) NOT NULL DEFAULT 'Untitled game',
  `description` varchar(1024) NOT NULL DEFAULT "",
  `price` float NOT NULL DEFAULT 0,
  `min_spec` int(11) DEFAULT NULL,
  `rec_spec` int(11) DEFAULT NULL,
  `publisher` int(11) DEFAULT NULL,
  `developer` int(11) DEFAULT NULL,
  `steamid` bigint(20) DEFAULT NULL,
  `release_date` datetime DEFAULT NULL,
  `downloads` int(11) NOT NULL DEFAULT 0,
  `wishlists` int(11) NOT NULL DEFAULT 0,
  PRIMARY KEY (`id`),
  KEY `developer_fk` (`developer`),
  KEY `min_spec_fk` (`min_spec`),
  KEY `publisher_fk` (`publisher`),
  KEY `rec_spec_fk` (`rec_spec`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8
COLLATE=utf8_hungarian_ci;

```

```
ALTER TABLE `games`  
  
  ADD CONSTRAINT `developer_fk` FOREIGN KEY (`developer`)  
  REFERENCES `companies` (`id`) ON DELETE SET NULL ON UPDATE  
  CASCADE,  
  
  ADD CONSTRAINT `min_spec_fk` FOREIGN KEY (`min_spec`)  
  REFERENCES `specifications` (`id`) ON DELETE SET NULL ON UPDATE  
  CASCADE,  
  
  ADD CONSTRAINT `publisher_fk` FOREIGN KEY (`publisher`)  
  REFERENCES `companies` (`id`) ON DELETE SET NULL ON UPDATE  
  CASCADE,  
  
  ADD CONSTRAINT `rec_spec_fk` FOREIGN KEY (`rec_spec`)  
  REFERENCES `specifications` (`id`) ON DELETE SET NULL ON UPDATE  
  CASCADE;
```

4.3.3.5 Libraries



A libraries egy many-to-many kapcsolótábla, mely meghatározza, hogy egy felhasználó mely játékokat birtokolja, valamint pár ehhez kapcsolódó adatot tárol.

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett
1	username 	varchar(32)	utf8_hungarian_ci		Nem	Nincs
2	game_id 	int(11)			Nem	Nincs
3	playtime	int(11)			Nem	0
4	is_favourite	tinyint(1)			Nem	0

```
CREATE TABLE IF NOT EXISTS `libraries` (  
  `username` varchar(32) NOT NULL,  
  `game_id` int(11) NOT NULL,  
  `playtime` int(11) NOT NULL DEFAULT 0,  
  `is_favourite` tinyint(1) NOT NULL DEFAULT 0,  
  UNIQUE KEY `username` (`username`,`game_id`),  
  KEY `lib_game_id_fk` (`game_id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8  
COLLATE=utf8_hungarian_ci;  
  
ALTER TABLE `libraries`  
  
  ADD CONSTRAINT `lib_game_id_fk` FOREIGN KEY (`game_id`)  
REFERENCES `games` (`id`) ON UPDATE CASCADE,  
  
  ADD CONSTRAINT `lib_username_fk` FOREIGN KEY (`username`)  
REFERENCES `users` (`username`) ON DELETE CASCADE ON UPDATE  
CASCADE;
```

4.3.3.6 Wishlists



A wishlists egy many-to-many kapcsolótábla, mely meghatározza, hogy mely játékok vannak egy adott felhasználó kívánságlistáján.

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett
1	username 	varchar(32)	utf8_hungarian_ci		Nem	Nincs
2	game_id 	int(11)			Nem	Nincs

```
CREATE TABLE IF NOT EXISTS `wishlists` (  
  `username` varchar(32) NOT NULL,  
  `game_id` int(11) NOT NULL,  
  UNIQUE KEY `username` (`username`,`game_id`),  
  KEY `wish_game_id_fk` (`game_id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8  
COLLATE=utf8_hungarian_ci;  
  
ALTER TABLE `wishlists`  
  
  ADD CONSTRAINT `wish_game_id_fk` FOREIGN KEY (`game_id`)  
REFERENCES `games` (`id`) ON DELETE CASCADE ON UPDATE  
CASCADE,  
  
  ADD CONSTRAINT `wish_username_fk` FOREIGN KEY (`username`)  
REFERENCES `users` (`username`) ON DELETE CASCADE ON UPDATE  
CASCADE;
```

4.3.3.7 Reviews



A reviews egy many-to-many kapcsolótábla, mely tartalmazza a játékok felhasználói értékeléseit.

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett
1	username 	varchar(32)	utf8_hungarian_ci		Nem	Nincs
2	game_id 	int(11)			Nem	Nincs
3	is_positive	tinyint(1)			Nem	Nincs
4	content	varchar(512)	utf8_hungarian_ci		Nem	Nincs

```
CREATE TABLE IF NOT EXISTS `reviews` (  
  `username` varchar(32) NOT NULL,  
  `game_id` int(11) NOT NULL,  
  `is_positive` tinyint(1) NOT NULL,  
  `content` varchar(512) NOT NULL,  
  UNIQUE KEY `username` (`username`,`game_id`),  
  KEY `rev_game_id_fk` (`game_id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8  
COLLATE=utf8_hungarian_ci;  
  
ALTER TABLE `reviews`  
  ADD CONSTRAINT `rev_game_id_fk` FOREIGN KEY (`game_id`)  
  REFERENCES `games` (`id`) ON DELETE CASCADE ON UPDATE  
  CASCADE,  
  ADD CONSTRAINT `rev_username_fk` FOREIGN KEY (`username`)  
  REFERENCES `users` (`username`) ON DELETE CASCADE ON UPDATE  
  CASCADE;
```


4.3.3.8 Carts

A carts egy many-to-many kapcsolótábla, mely meghatározza, hogy mely játékok vannak egy adott felhasználó kosarában.

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett
1	username 	varchar(20)	utf8_hungarian_ci		Nem	Nincs
2	game_id 	int(11)			Nem	Nincs
3	quantity	int(11)			Nem	Nincs

```
CREATE TABLE IF NOT EXISTS `carts` (  
  `username` varchar(20) NOT NULL,  
  `game_id` int(11) NOT NULL,  
  `quantity` int(11) NOT NULL,  
  KEY `username` (`username`),  
  KEY `game_id` (`game_id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8  
COLLATE=utf8_hungarian_ci;  
  
ALTER TABLE `carts`  
  
  ADD CONSTRAINT `carts_ibfk_1` FOREIGN KEY (`username`)  
REFERENCES `users` (`username`) ON DELETE CASCADE ON UPDATE  
CASCADE,  
  
  ADD CONSTRAINT `carts_ibfk_2` FOREIGN KEY (`game_id`)  
REFERENCES `games` (`id`) ON DELETE CASCADE ON UPDATE  
CASCADE;
```

4.4 Szerepkörök

4.4.1 Vendég felhasználó

Amikor a felhasználó megnyitja az alkalmazást, és korábban nem jelentkezett be, akkor vendég felhasználóként tekinti azt meg, amíg be nem jelentkezik vagy regisztrál.

4.4.1.1 *Vendég felhasználó jogosultságai az asztali alkalmazáson*

- Bejelentkezés
- Regisztráció

4.4.1.2 *Vendég felhasználó jogosultságai a weblapon*

- Bejelentkezés
- Regisztráció
- Kínálat böngészése
- Játékok oldalának megtekintése

4.4.2 Felhasználó

Miután a vendég felhasználó bejelentkezett, a felhasználói szerepkörbe lép. Ebben a szerepkörben lehetősége nyílik használni az áruház eddig elérhetetlen funkcióit.

4.4.2.1 *Felhasználó jogosultságai az asztali alkalmazáson*

- Kínálat böngészése
- Játékok oldalának megtekintése
- Kívánságlista megtekintése
- Könyvtár megtekintése
- Profil megtekintése

4.4.2.2 *Felhasználó jogosultságai a weblapon*

- Kínálat böngészése
- Játékok oldalának megtekintése
- Kívánságlista megtekintése
- Könyvtár megtekintése
- Profil megtekintése és szerkesztése
- Játékok kosárba helyezése
- Kosár tartalmának megvásárlása

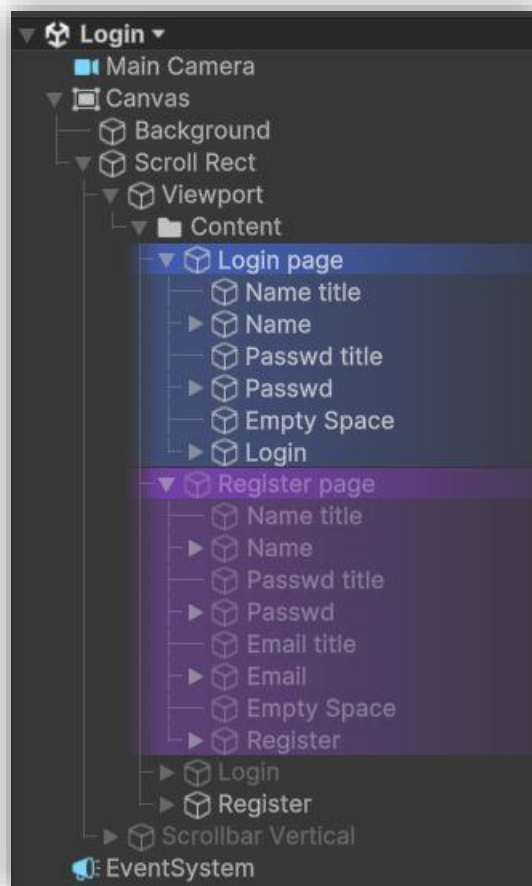
4.5 Alkalmazások szerkezete

4.5.1 Asztali alkalmazás

Az asztali alkalmazás két scene-re van osztva, melyekhez 1-1 szerepkör van rendelve.

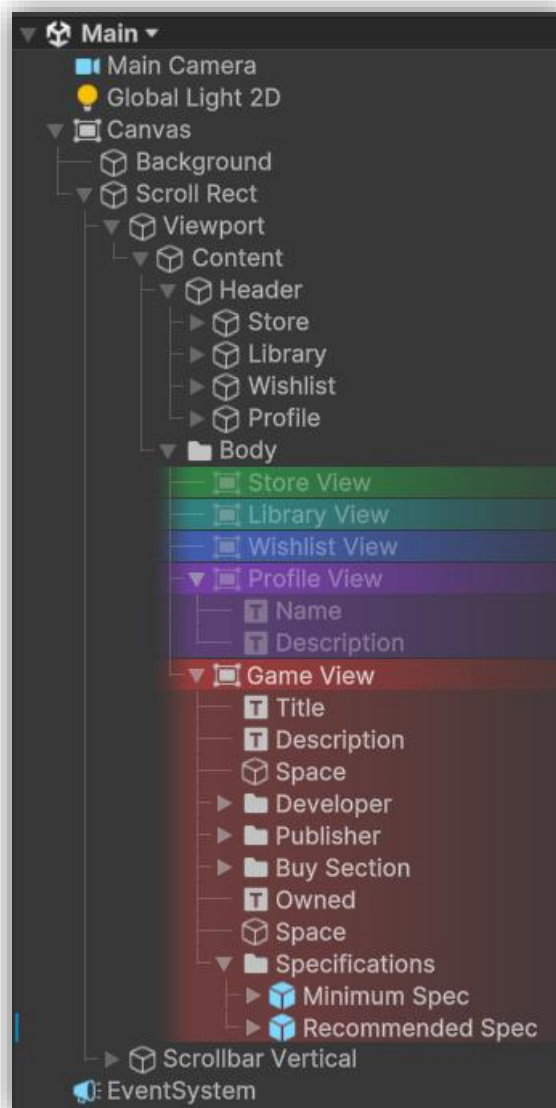
4.5.1.1 Login scene

Az alkalmazás indításakor az első dolog, amit a felhasználó lát a Login scene. Ezen a scene-en a felhasználó még nem jelentkezett be, így vendég felhasználói szerepkörben van. Ez a scene egy bejelentkezési és egy regisztrálási felületet kínál, valamint két gombot, mely segítségével a kettő között válthatsz. A scene fő szerkezetét egy scrollrect-ben lévő vertical layout adja. A bejelentkezés vagy regisztráció űrlap kitöltése után a program kapcsolatot létesít az API-val, megteszi a szükséges lekérdezéseket, majd átirányít a Main scene-re.



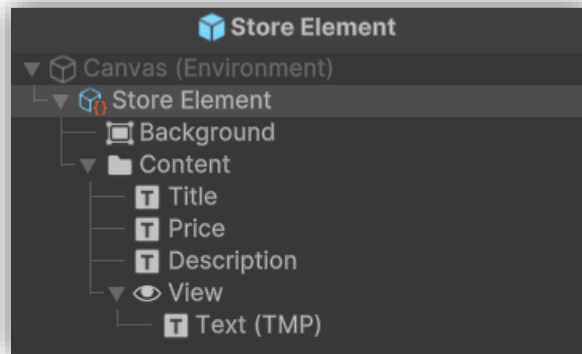
4.5.1.2 Main scene

A bejelentkezés után az alkalmazás átirányít a Main scene-re, ahol a felhasználó, immár felhasználói szerepkörben, hozzáférhet az áruházhoz. Ez a scene 5 különböző felületet kínál: Store view, Library view, Wishlist view, Profile view, és Game view. Ezen túl az oldal tetején található egy navigációs sáv, mellyel a különböző felületek között válthat a felhasználó. Hasonlóan a Login scene-hez, itt is egy scrollrect-ben lévő vertical layout biztosítja a scene fő szerkezetét.



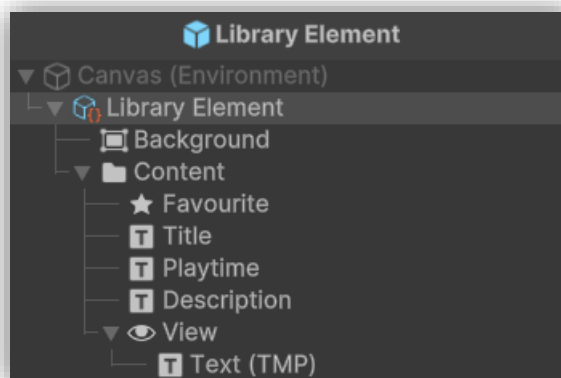
4.5.1.2.1 Store view

A Store view felületen a felhasználó böngészheti az áruház kínálatát. A felület tartalma runtime generálódik az API-tól kapott adatok alapján. Minden játéknak készül egy külön kártya az alábbi Store Element prefab alapján. Ezeken a kártyákon megtekinthető a játék címe, ára, leírása, valamint egy gomb, ami átirányít a játékhoz tartozó Game view-ra. Ezek a kártyák végül függőlegesen egymás alá rendezésre kerülnek.



4.5.1.2.2 Library view

A Library view felületen a felhasználó megtekintheti az általa bírtokolt játékokat. Ez a felület szintén runtime generált az API alapján. A Store view-hoz hasonlóan minden játék saját kártyát kap, ám ebben az esetben a Library Element prefab mintájára. A Store Element mintához képest ezen nincs rajta a játék ára, ám fel van tüntetve a felhasználó játék ideje, valamint egy csillag a bal felső sarokba, ha fel van véve a kedvencek közé. A kártyák végül függőlegesen egymás alá kerülnek, majd sorrendbe állnak ABC sorrend szerint. A kedvencek mindig a lista elején vannak.



4.5.1.2.3 Wishlist view

A Wishlist view felület nagyon hasonló a Store view felülethez. A felépítése megegyezik a Store view felületével, valamint működésben csak annyiban különbözik, hogy nem a teljes áruház tartalmát jeleníti meg, csak a felhasználó által a kívánságlistához adott játékokat.

4.5.1.2.4 Profile view

A Profile view felületen a felhasználó megtekintheti a saját profilját. A felület felépítése előre el van készítve, runtime csak az adatok kerülnek kicserélésre.

4.5.1.2.5 Game view

A Game view az egyetlen olyan felület, mely egy játékhoz van kötve. A boltban minden játék kártyáján található egy gomb, mely megnyitja a játékhoz tartozó Game view-t. A felépítése előre el van készítve, az adatok kerülnek kicserélésre akárhányszor a felhasználó megnyitja azt. A felületen többek között megjelenik a játék minimum és ajánlott specifikációja, valamint egy gomb, mely átirányít a weblapra a vásárláshoz.

4.5.1.3 Kódolás

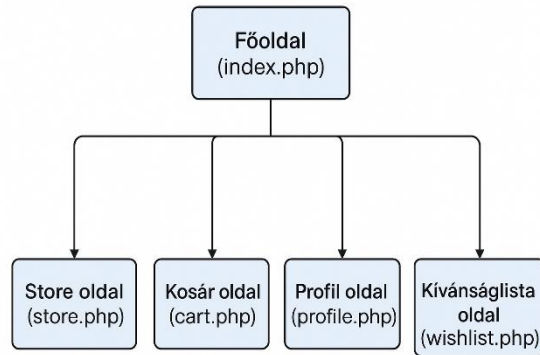
Az asztali alkalmazásban minden adatbázis tábla kapott egy interfész osztályt, mely le van programozva, hogy az adott adatot, hogy lehet lekérni, valamint beállítani, így közvetítőként működve az API és az asztali alkalmazás többi funkciója között. Ezen túl minden felületnek, valamint prefabnak van egy saját scriptje, mely konfigurálja az adott felületet vagy prefabot, hogy megegyezzen az adatokkal.

4.5.1.4 Konfiguráció

A könnyű konfigurálás érdekében az asztali alkalmazás a weblapra és az API-ra egy referencia fájlal hivatkozik. Az alkalmazásban minden script ezeken keresztül tud kommunikálni velük, így ha az elérési útvonaluk megváltozna, azt elég lenne egy helyen átállítani.

4.5.2 Weboldal

A weboldal szerkezete alapvetően a főoldalra épül, onnan érhető el a többi oldal, ahogy a kapcsolati ábra is mutatja



4.5.2.1 Login

A weboldal megnyitásakor az első, amit a felhasználó lát a bejelentkezési oldal. Itt a felhasználó a még nem jelentkezett be, így vendég felhasználói szerepkörben van. Ez az oldal egy bejelentkezési és egy regisztrációs felületet kínál, valamint két gombot, mely segítségével a kettő között válthatsz. A bejelentkezés vagy regisztráció űrlap kitöltése után a program kapcsolatot létesít az API-val, megteszi a szükséges lekérdezéseket, majd átirányít a főoldalra.

4.5.2.2 Főoldal

A főoldal a webáruház kezdőlapja, amely automatikusan betöltődik a bejelentkezést követően. Ez az oldal dinamikusan listázza a termékeket (játékokat), amelyeket a háttérrendszerből tölt be. A fejlesztés során itt kerülnek megjelenítésre a legújabb, ajánlott vagy akciós játékok is.

Az oldal felépítése:

- Betölti a központi fejléct.
- Meghívja a stíluslapot (styles.css) a megjelenéshez.
- Kapcsolódik a script.js fájlhoz a dinamikus műveletekhez.
- A termékek listázása PHP ciklus segítségével történik, ami adatbázisból olvassa ki az adatokat (feltételezhetően, mivel az adatbáziskezelés nincs most nálam).

4.5.2.3 Header

A *header.php* fájl egy központi fejléct biztosít, amely minden oldalon megjelenik. Ez a fájl tartalmazza a főbb navigációs elemeket, például a főoldalra mutató linket, a felhasználói profil elérését, a kosárhoz és kívánságlistához való gyors hozzáférést, valamint a kijelentkezés lehetőségét.

4.5.2.4 Kosár oldal

A kosár oldal felel a felhasználó által kiválasztott játékok összesítéséért és megjelenítéséért.

Az oldal felépítése:

- Tartalmazza a fejléct, így a navigáció minden oldalról elérhető.
- Megjeleníti a kosárban található termékeket, azok árát és a végösszeget.
- Lehetőséget ad a kosárban lévő tételek mennyiségének módosítására vagy eltávolítására.
- Tovább lépési lehetőség a fizetés felé, amennyiben a projekthez tartozik backend fizetési modul.

4.5.2.5 Store oldal

A store oldal célja a termékkatalógus megjelenítése, részletesebb szűrési és keresési lehetőségekkel.

Az oldal felépítése:

- Tartalmazza a navigációs fejléct.
- Megjeleníti az összes elérhető terméket kategóriákra bontva, vagy szűrhető módon.
- A felhasználó termékcímkékre vagy szűrőkre kattintva szűkítheti a találatokat.
- Minden terméknel megjelenik a kosárba vagy a kívánságlistára helyezés lehetősége.

4.5.2.6 Profil oldal

A profil oldal a bejelentkezett felhasználó személyes adatait és tevékenységeit jeleníti meg.

Az oldal felépítése:

- Megjeleníti a felhasználói adatokat: név, e-mail cím, regisztráció dátuma stb.
- Listázza a felhasználó vásárlási előzményeit, ha ilyen funkció implementálva van.
- Lehetőség van profiladatok szerkesztésére.

4.5.2.7 Kívánságlista oldal

A kívánságlista oldal lehetőséget ad a felhasználóknak arra, hogy elmentsék a később megvásárolni kívánt játékokat.

Az oldal felépítése:

- Tartalmazza a navigációs fejléct, hogy könnyen navigálható legyen.
- Megjeleníti a kívánságlistára helyezett játékokat, azok aktuális áraival.
- Lehetőség van a termékek kosárba helyezésére vagy eltávolítására a kívánságlistáról.

4.5.2.8 Kódolási irányelvek

A projekt PHP alapú, HTML, CSS és JavaScript használatával. Az alábbi szabályokat és ajánlásokat követi:

4.5.2.8.1 Backend (PHP)

- Az oldal PHP fájljai szerveroldali feldolgozást végeznek.
- Központi fejléc (header.php) minden oldalon dinamikusan betöltésre kerül.
- A dinamikus tartalom (termékek listázása, kosár tartalma, kívánságlista stb.) PHP ciklusokkal jelenik meg.
- A biztonság érdekében használt:
 - **Prepared Statements** SQL lekérdezésekhez (pl. PDO vagy MySQLi)
 - **Session alapú hitelesítés** a felhasználói bejelentkezésekhez
 - **Input validáció** a formoknál

4.5.2.8.2 Frontend (HTML, CSS, JavaScript)

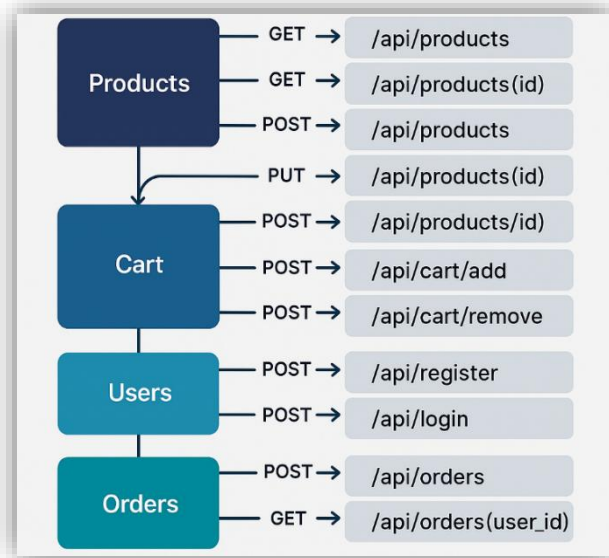
- A megjelenést a styles.css fájl határozza meg.
- Az oldalak reszponzív felépítésűek, így mobileszközökön is jól használhatók.
- Interaktív elemekhez (kosár frissítés, termék eltávolítás stb.) a script.js fájl használatos.

4.5.3 Rest API

A diagram a REST API működését mutatja be lépésről lépésre.

Az API-t a frontend alkalmazás hívja meg, például amikor a felhasználó termékeket szeretne böngészni, rendelést leadni, vagy a kosarát frissíteni.

Elsőként a Frontend alkalmazás kezdeményezi a kérést az API felé.



Ez lehet például:

- GET kérés a termékek listázásához,
- POST kérés a kosár frissítéséhez,
- vagy POST kérés felhasználói regisztrációhoz.

A kérés az API rétegen keresztül halad, amely ellenőrzi, hogy a kérés hitelesített-e, ha szükséges.

Az API továbbítja a kérést az Adatbázis kezelő rétegnek, amely elvégzi az adatbeolvasást vagy módosítást. Például, ha terméklistát kér a felhasználó, a rendszer beolvassa a termékek adatait az adatbázisból. Miután az adatbázis kiszolgáltatta a kérést, a válasz visszakerül az API réteghez, amely formázza a választ JSON struktúrába, és elküldi a frontend számára. A Frontend alkalmazás ezt a választ feldolgozza, és megjeleníti a felhasználónak, például frissíti a terméklistát, vagy megerősíti a kosár frissítését.

A diagram hangsúlyozza, hogy az API a rendszer központi kapcsolódási pontja, amely összeköti a frontend felhasználói felületet az adatbázis háttérfolyamataival, így biztosítva a gördülékeny adatcserét.

4.6 Funkciók bemutatása

4.6.1 Játéklista

A játéklista az áruház fő oldala, ahol a teljes kínálat megtekinthető. Itt a felhasználó alapvető információkat kaphat minden játékról, mely alapján el tudja dönteni, hogy az adott játék érdekli-e. A játéklista ezen túl átirányíthat a játék saját oldalára, ahol részletesebb információk találhatók a játékról.

4.6.2 Kívánságlista

A kívánságlista a játéklista szűrt változata. Itt csak azok a játékok jelennek meg, amiket a felhasználó korábban kiválasztott. Ezen túl minden egyéb funkciójában azonos a játéklissával.

4.6.3 Könyvtár

A könyvtárban a felhasználó láthatja az általa birtokolt játékokat, és megtekintheti a játékhoz fűződő saját adatait.

4.6.4 Profil

A profil segítségével a felhasználó megtekintheti, valamint módosíthatja a saját adatait.

4.6.5 Játék oldal

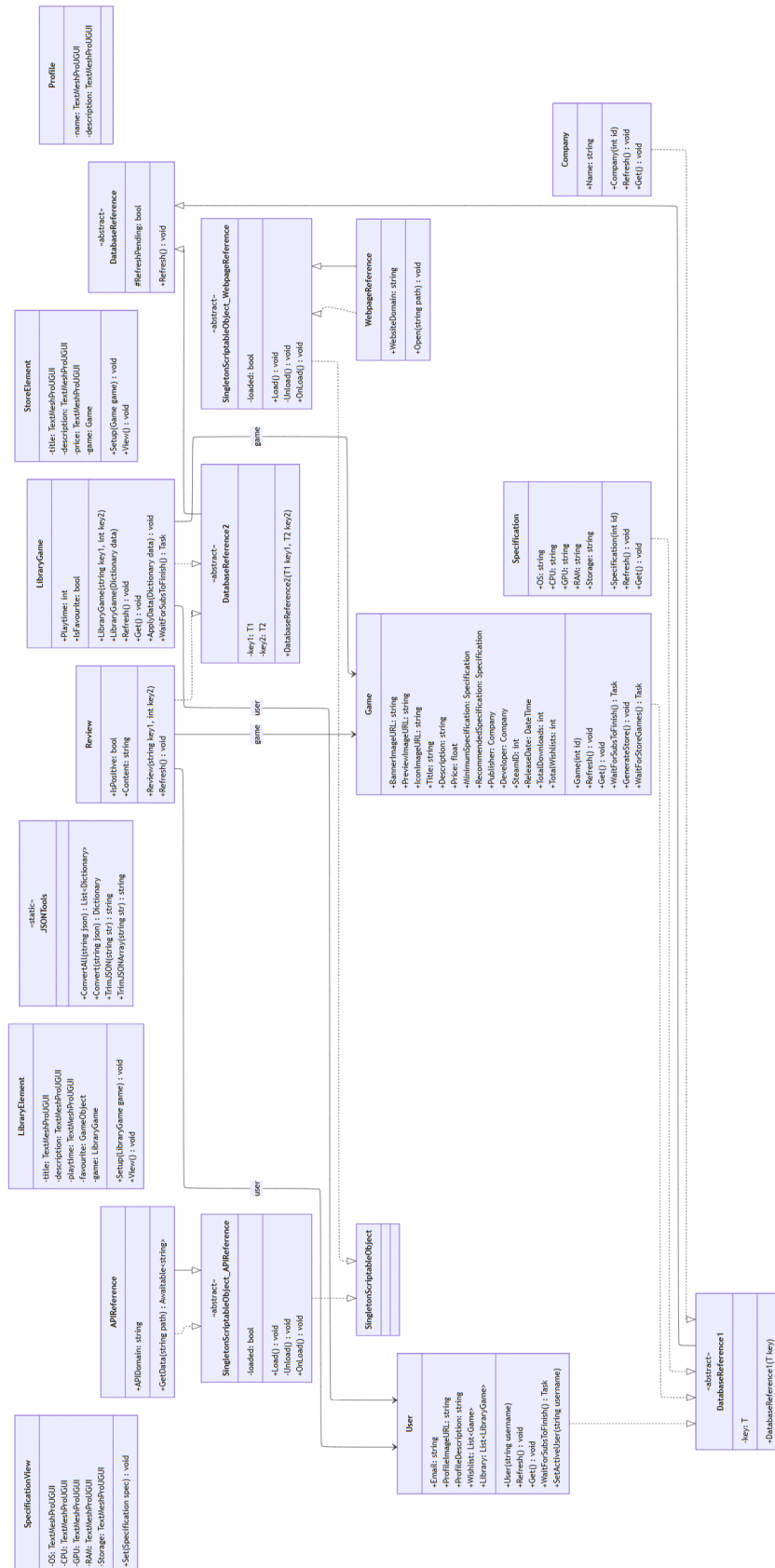
Minden játékhoz tartozik egy játék oldal. Ezeken az oldalakon a felhasználó részletes leírást kaphat az adott játékhoz, valamint vásárlást is kezdeményezhet.

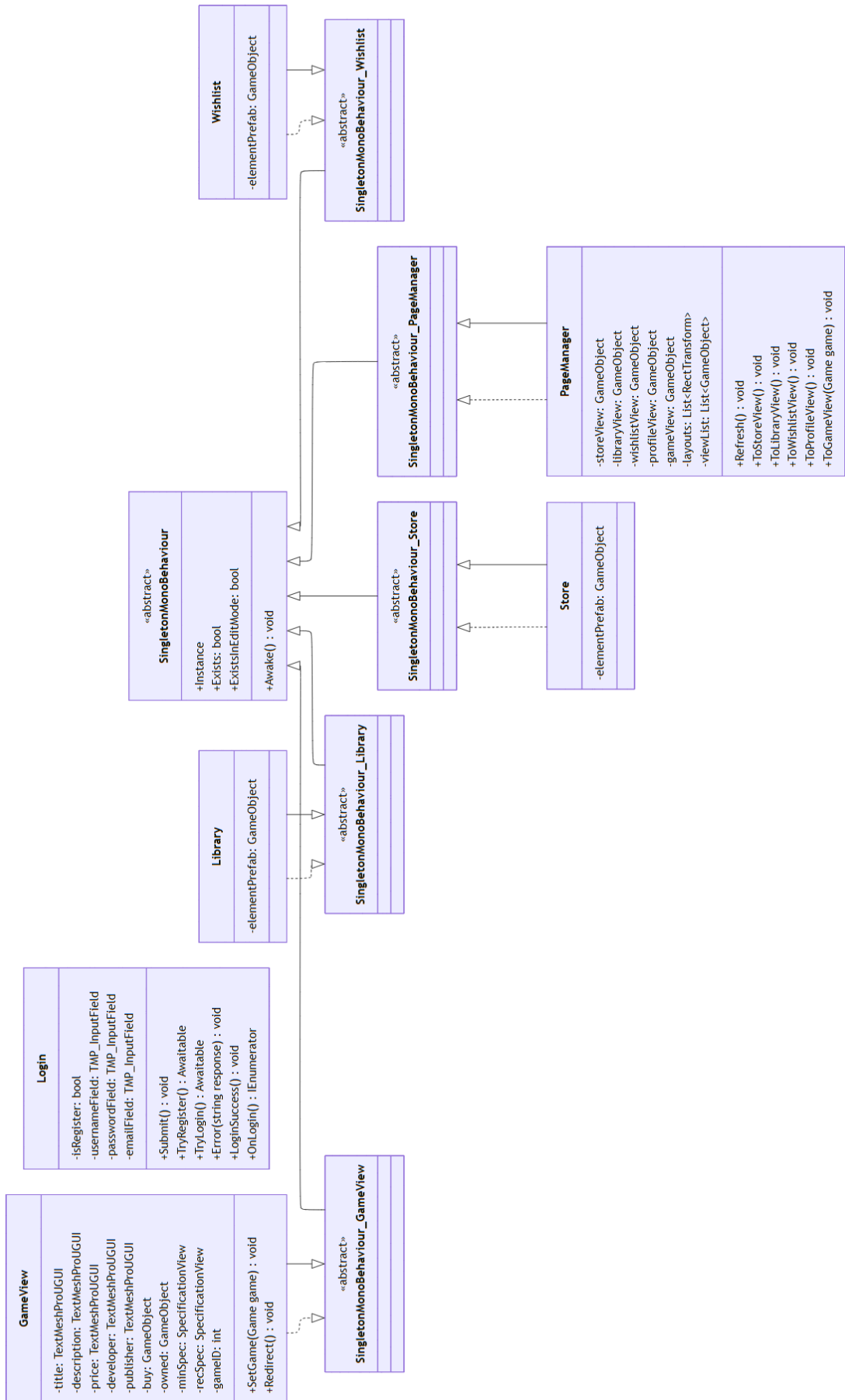
4.6.6 Kosár

A kosár segítségével a felhasználó kezelheti a megvásárolni kívánt termékek listáját, valamint véglegesítheti a vásárlást.

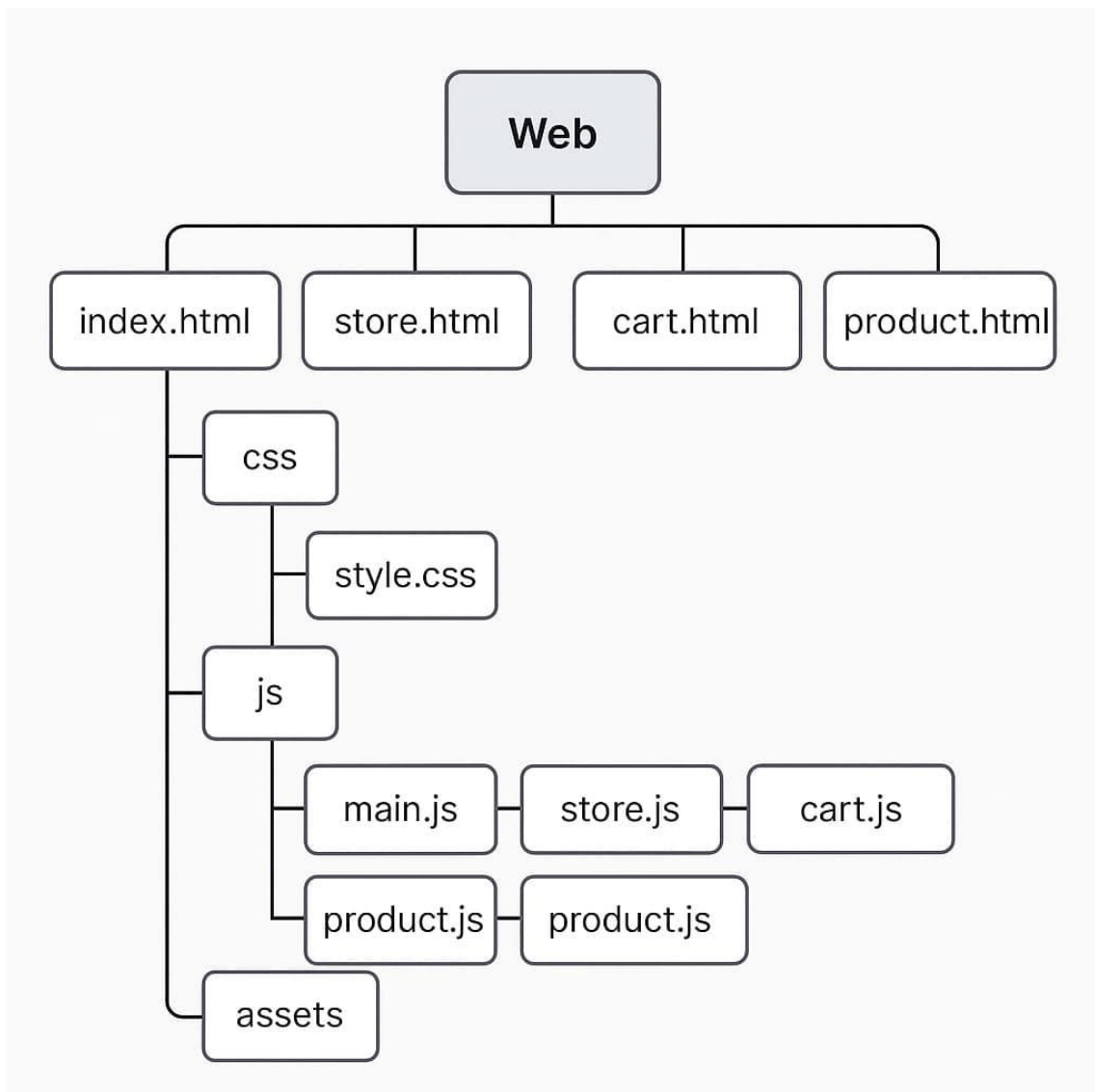
4.7 Kapcsolati diagram

4.7.1 Asztali alkalmazás

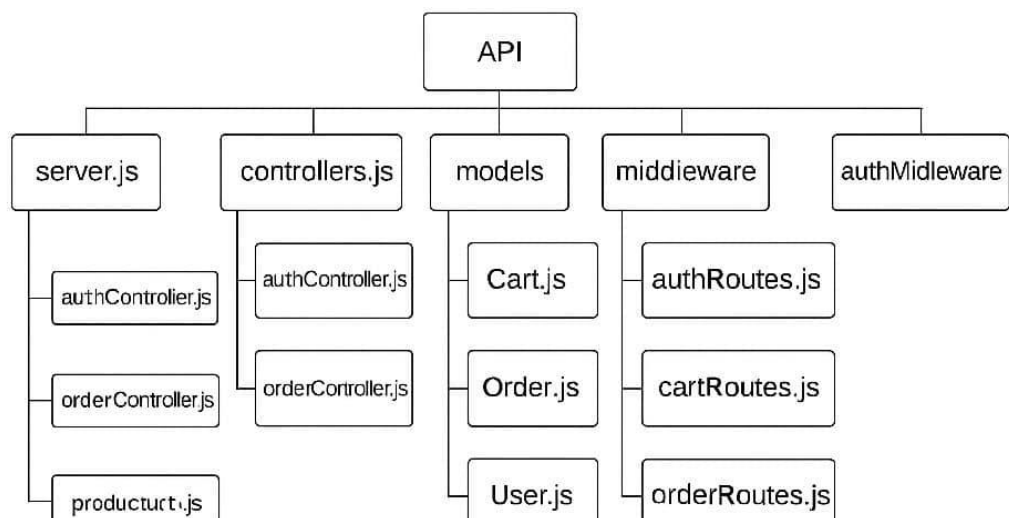




4.7.2 Web



4.7.3 Rest API



4.8 Bemeneti értékek ellenőrzése

Az API implementációja során gondosan ügyeltünk a bemeneti adatok ellenőrzésére, hogy elkerülhető legyen a nem kívánt adatbevitel, a hibák vagy biztonsági kockázatok (pl. injection támadások).

Végpont (API kódot)	Paraméter / Mező	Típus	Kötelező	Ellenőrzések / Megkötések
POST /register	username	String	Igen	Minimum 3 karakter, nem üres
	email	String (Email)	Igen	Valós e-mail formátum ellenőrzése
POST /login	email	String	Igen	Érvényes e-mail formátum
	password	String	Igen	Érvényes e-mail formátum
POST /products	name	Number	Igen	Érvényes email formátum
	price	String	Igen	Nem üres érték
POST /cart/add	description	Number	Nem	Pozitív szám 100 karakter
POST /order/checkout	stock	Number	Igen	Max 500 karakter
	productId	String (ID)	Igen	Érvényes termék azonosító
POST /order/checkout	quantity	Number	1	Minimum érték: 1
	cardid	String (ID)	Igen	Csak előre definiált értékek pl.: "card" "paypal"
POST /products/review	rating	Number	Igen	1-től 5-ig terjedő értékek
	comment	String	Nem	Max 300 karakter

4.8.1 Hitelesítési adatok ellenőrzése

A regisztráció és bejelentkezés során az API ellenőrzi:

- A felhasználónév és email-cím meglétét és formátumát.
- Az email cím érvényességét.
- A jelszó minimális hosszát és komplexitását, hogy biztonságos legyen. (Például minimum 6-8 karakter, tartalmazzon számot és betűt.)

A jelszavakat bcryptjs könyvtárral biztonságosan hashelik, így azok titkosítva kerülnek tárolásra az adatbázisban.

4.8.2 JWT token ellenőrzés

A védett végpontok eléréséhez a kérés fejlécében kötelezően szerepelnie kell a Authorization: Bearer [token] sorozatnak.

A authMiddleware.js ellenőrzi:

- A token meglétét.
- A token érvényességét, lejáratát idejét.
- Ha a token hiányzik vagy érvénytelen, a rendszer hibát ad vissza, és nem engedélyezi a műveletet.

4.8.3 Kosár és rendelés műveletek bemeneti ellenőrzése

Amikor a felhasználó új terméket ad a kosárhoz vagy rendelést hoz létre:

- Ellenőrzi, hogy a termékazonosító (game_id) helyes és létező legyen az adatbázisban.
- Ellenőrzi a megadott mennyiség típusát és értékét, például, hogy pozitív egész szám legyen.

- Ha bármilyen adat hiányzik vagy hibás, a rendszer 400-as hibakóddal (Bad Request) válaszol, és részletes hibaüzenetet ad vissza.

4.8.4 Termékek lekérdezése

A terméklisták és egyes termékek adatainak lekérésénél:

- Az egyedi termékazonosító ellenőrzése történik (pl. MongoDB ObjectId formátum).
- Ha a kért termék nem található, az API megfelelő hibakódot és üzenetet küld vissza.

4.8.5 Általános hibakezelés

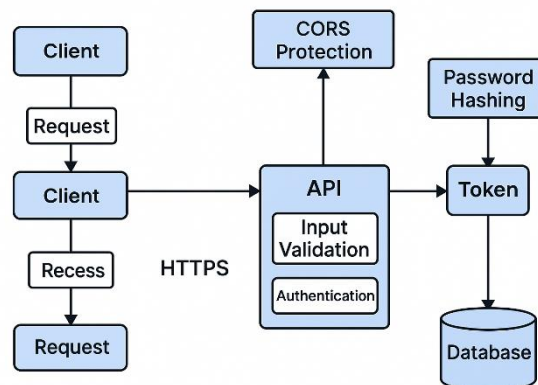
A teljes API-ban egységes hibakezelés működik:

- Hibás bemeneti adat esetén minden végpont részletes hibaüzenetet ad vissza, megkönnyítve a frontend oldali hiba-kezelést.
- Nem várt hibák esetén az API 500-as serverhibát küld.
- A hibák kezelése fejlesztőbarát módon van kialakítva, így a hibák gyorsan diagnosztizálhatók.

4.9 Biztonsági intézkedések

A projekt több szinten alkalmaz biztonsági mechanizmusokat, hogy megvédje a rendszert az illetéktelen hozzáféréstől, az adatszivárgástól és a támadásoktól.

4.9.1 JWT alapú hitelesítés (JSON Web Token)



- A felhasználó hitelesítése JWT token segítségével történik.
- A token a bejelentkezés után kerül generálásra, és minden további kérésnél a fejlécben kell megadni.
- A token tartalmazza a felhasználó azonosítóját és lejáratát, így minden kérésnél a rendszer ellenőrzi, hogy a token:
 - Helyesen lett aláírva.

- Nem járt-e le.
 - Valóban egy létező felhasználót azonosít.
- Ha a token érvénytelen vagy hiányzik, a rendszer elutasítja a kérést.

4.9.2 Jelszóhash-elés (bcryptjs)

- A felhasználók jelszavait nem nyers szövegként tároljuk, hanem bcrypt algoritmussal hash-elve.
- Még sikeres adatbázis-hozzáférés esetén is olvashatatlanok a jelszavak.
- A jelszavak validálása összehasonlító hash ellenőrzéssel történik, nem pedig szöveges összevetéssel.

4.9.3 Adatvalidálás a bemeneti oldalon

- Az API bemeneti adatokat minden végpontnál szigorúan ellenőrzi.
- Ellenőrzi a kötelező mezőket, adattípusokat, formátumokat (pl. email formátum, pozitív számok).
- Hibás vagy hiányzó adat esetén a rendszer nem dolgozza fel a kérést, és részletes hibaüzenetet ad vissza.

4.9.4 Middleware védelem

- Az alkalmazás köztes réteget (middleware) használ az érzékeny végpontok védelmére.
- Például a kosár, rendelés vagy felhasználói adatok kezeléséhez middleware ellenőrzi a felhasználó azonosítóját a JWT tokenből.

4.9.5 HTTPS használat ajánlott

- A dokumentáció és a fejlesztési irányelv javasolja a HTTPS protokoll használatát az adatkommunikáció titkosítására.
- Ez megakadályozza, hogy a tokenek vagy bármely más érzékeny adat lehallgatásra kerüljön a hálózaton keresztül.

4.9.6 CORS védelem

- A projekt kezeli a CORS (Cross-Origin Resource Sharing) szabályozását.
- Csak engedélyezett domeinekről érkező kéréseket fogad el, csökkentve a cross-site támadások esélyét.

4.9.7 Hibakezelés és információ visszatartás

- A rendszer nem árul el túl sok információt a hibaüzenetekben, így a támadók nem tudnak érzékeny részleteket megismerni (például adatbázis struktúrát vagy érzékeny háttér információt).

4.9.8 Session kezelés

- Mivel JWT-t használ, nincs klasszikus szerveroldali session kezelés. A tokenek stateless módon működnek, ami csökkenti a szerver oldali támadások felületét.

4.9.9 Időtartam alapú token lejárat

- A generált tokenek lejáratí idővel vannak ellátva.
- Ha a felhasználó nem frissíti a token, a hozzáférés automatikusan megszűnik a megadott időintervallum után.

5 Felhasználói dokumentáció

5.1 Rendszerkövetelmények

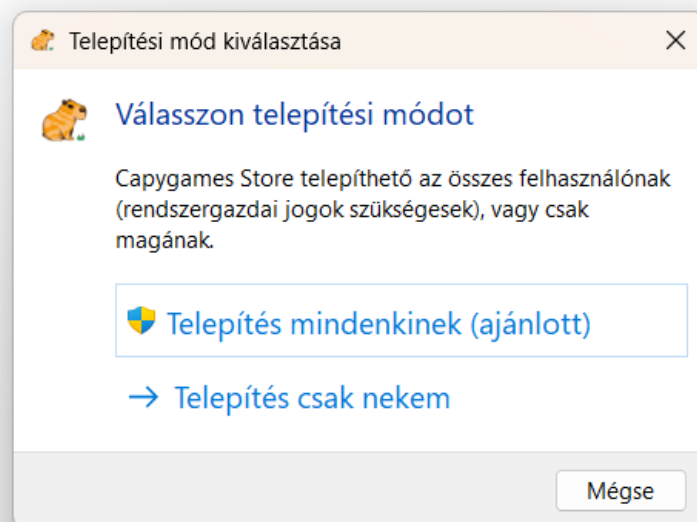
5.1.1 Asztali alkalmazás rendszerkövetelményei

- **Operációs rendszer:** Windows 10 vagy újabb
- **Processzor:** x86/x64 processzor SSE2 utasításkészlet támogatással vagy ARM64
- **Videókártya:** DirectX 10/11/12 kompatibilis videokártya
- **Tárhely:** 150MB
- **Egyéb:** Internetkapcsolat szükséges

5.2 Telepítési útmutató

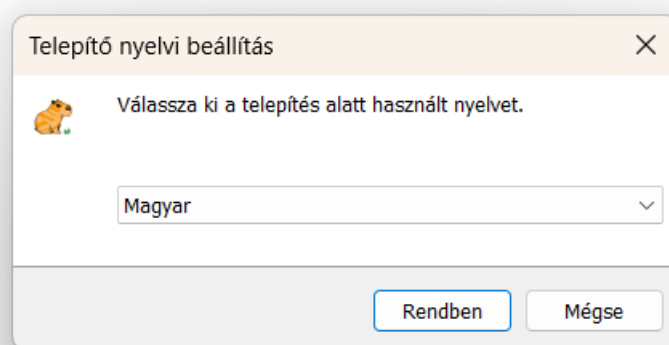
5.2.1 Asztali alkalmazás telepítése

Az asztali alkalmazás egy telepítő futtatása segítségével könnyen telepíthető. A telepítő elindítása után a következő ablak fogad:

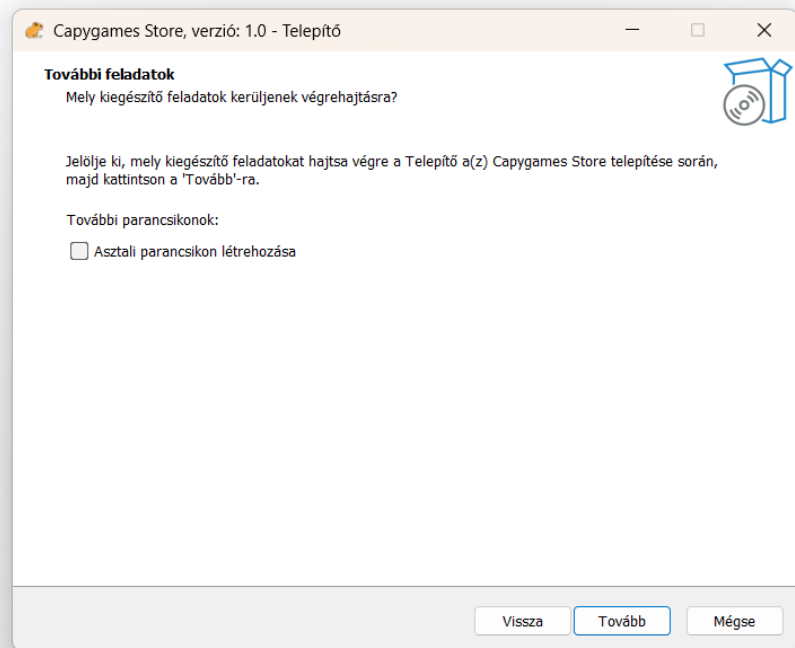
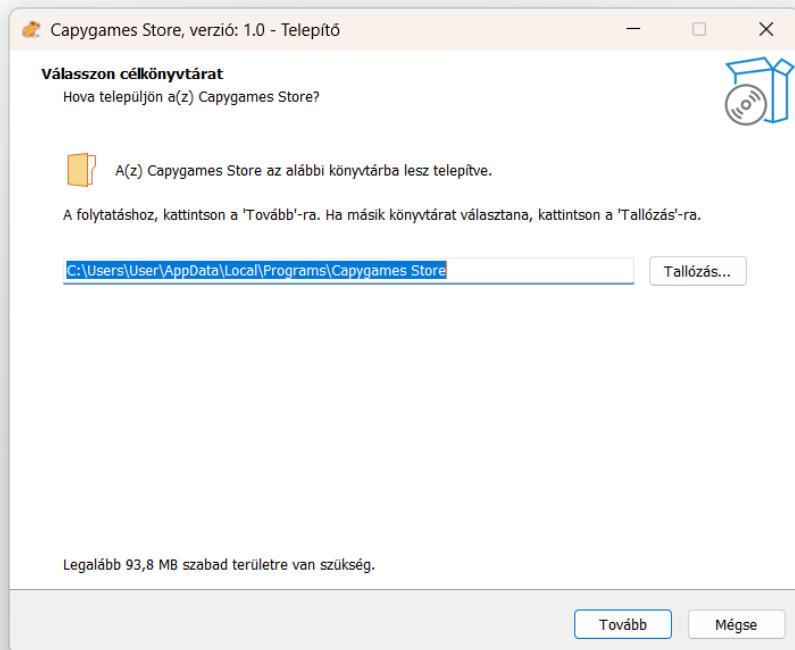


Ezen az ablakon kiválaszthatjuk, hogy minden felhasználó számára telepítse, vagy csak azon felhasználó számára, aki futtatta. Amennyiben a „Telepítés mindenkinek” opciót választjuk, a telepítő ehhez adminisztrátori jogosultságot fog kérni.

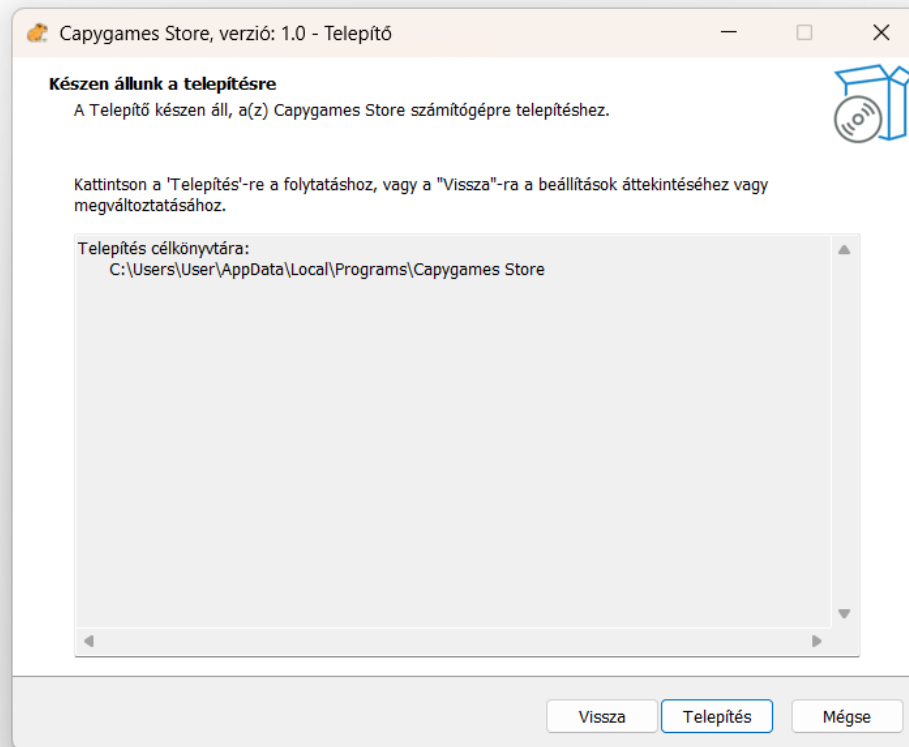
Ez után a telepítő megkérdezi, milyen nyelven szeretnénk folytatni a telepítést. Ez csak a telepítő által használt nyelvet befolyásolja, a telepített program nyelvét nem. A telepítéshez elérhető nyelvek az angol és a magyar.



Ezután a telepítő megkérdezi, hogy hová telepítse az asztali alkalmazás fájljait, valamint hogy készítsen-e asztali parancsikont.



Végül a telepítő összegzi a korábban megadott beállításokat, és megkérdezi, hogy biztosak vagyunk-e a telepítésben:



Amennyiben itt a „Telepítés” gombra nyomunk, az asztali alkalmazás telepítésre kerül.

5.2.2 Szerverkörnyezet telepítése

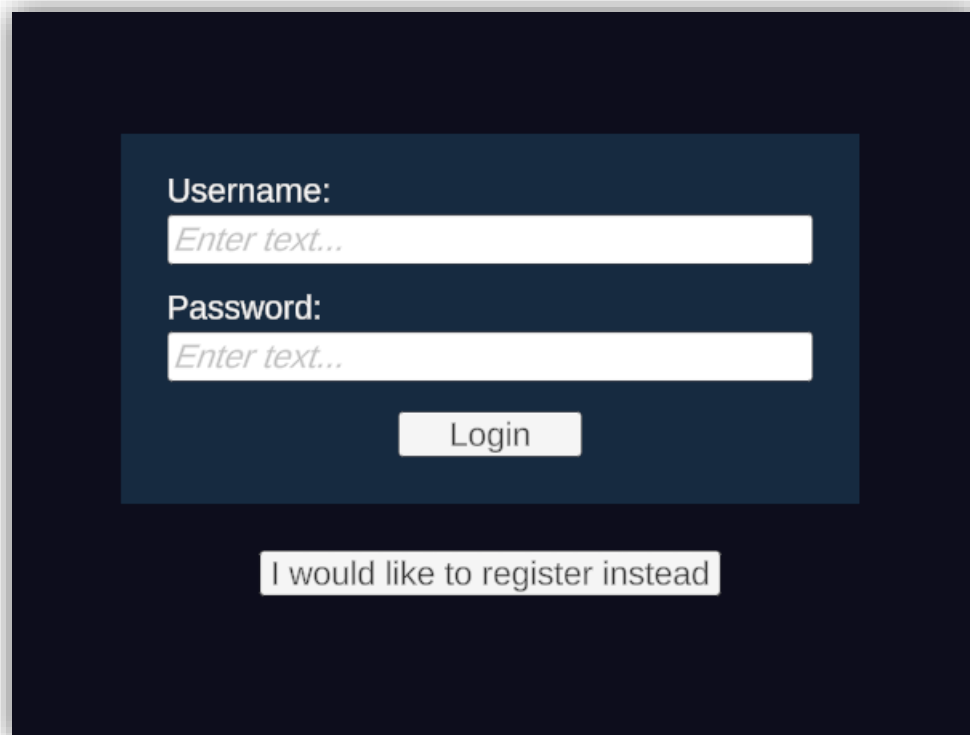
A szervert a felhasználó rendeltetésszerűen az interneten keresztül érheti el, így telepítésre nincs szükség, ám ha tesztelési célból mégis ilyet szeretne tenni, akkor ehhez a XAMPP programot ajánljuk. A weblap GitHub repository tartalmát a htdocs/Web, míg az API repository tartalmát a htdocs/API mappába kell helyezni a rendeltetésszerű működéshez. Ez után a XAMPP segítségével el kell indítani az Apache és MySQL szolgáltatásokat, és a database.sql fájlt beimportálni az adatbáziskezelőbe a phpMyAdmin segítségével. Ez után a szerver készenáll. Fontos, hogy így a felhasználó nem éri el a központi szerveret, és csak egy saját tesztverziót telepít, melyen a változások nincsenek kihatással a központi szerverekre.

5.3 Felületek bemutatása

5.3.1 Asztali alkalmazás felületei

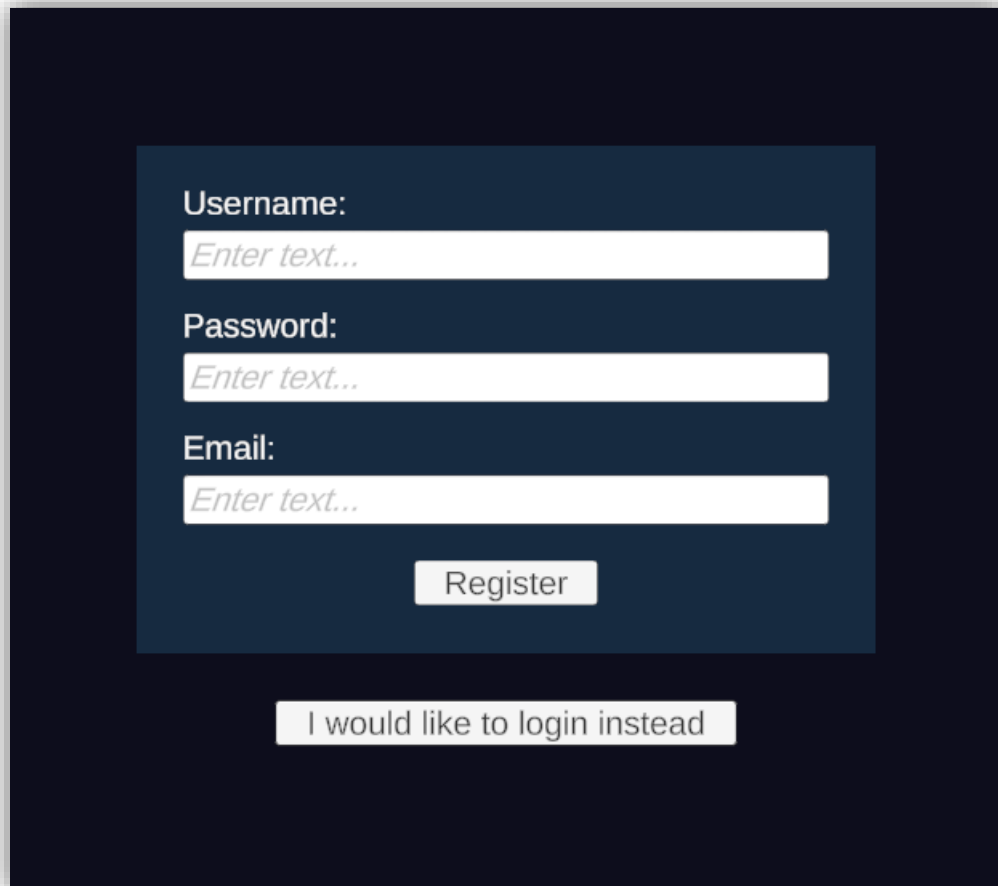
5.3.1.1 Bejelentkezés

A bejelentkezési felületen a felhasználó beírhatja a felhasználónevét a felső bementi mezőbe, valamint a jelszavát az alsó bemeneti mezőbe, majd a „Login” gombot megnyomva bejelentkezést kezdeményezhet. Amennyiben inkább regisztrálna, az „I would like to register instead” gombra nyomva átválthat a regisztrációs felületre.

A screenshot of a login form on a dark blue background. The form is centered and consists of a dark blue rectangular box containing two white text input fields. The first field is labeled 'Username:' and the second is labeled 'Password:'. Both fields have a light gray placeholder text 'Enter text...'. Below the password field is a white button with the text 'Login'. Below the entire blue box is a white button with the text 'I would like to register instead'.

5.3.1.2 Regisztráció

A regisztrációs felületen a felhasználó beírhatja a kiválasztott felhasználónevet a felső bementi mezőbe, az választott jelszavát a középső bemeneti mezőbe, valamint az email címét az alsó bemeneti mezőbe, majd a „Register” gombot megnyomva regisztrációt kezdeményezhet. Amennyiben inkább regisztrálna, az „I would like to login instead” gombra nyomva átválthat a bejelentkezési felületre.

The image shows a registration form on a dark blue background. The form is contained within a lighter blue rectangular area. It features three text input fields, each with a label above it: 'Username:', 'Password:', and 'Email:'. Each input field has a light gray placeholder text that reads 'Enter text...'. Below the input fields is a white button with the text 'Register'. At the bottom of the form area is another white button with the text 'I would like to login instead'.

5.3.1.3 Bolt

A bolt felületen a felhasználó megtekintheti az áruház teljes kínálatát, láthatja a játékok leírását, árát, valamint kiválaszthatja őket részletesebb megtekintésre. Ezen túl az oldal tetején található navigációs sávon átválthat más felületekre.



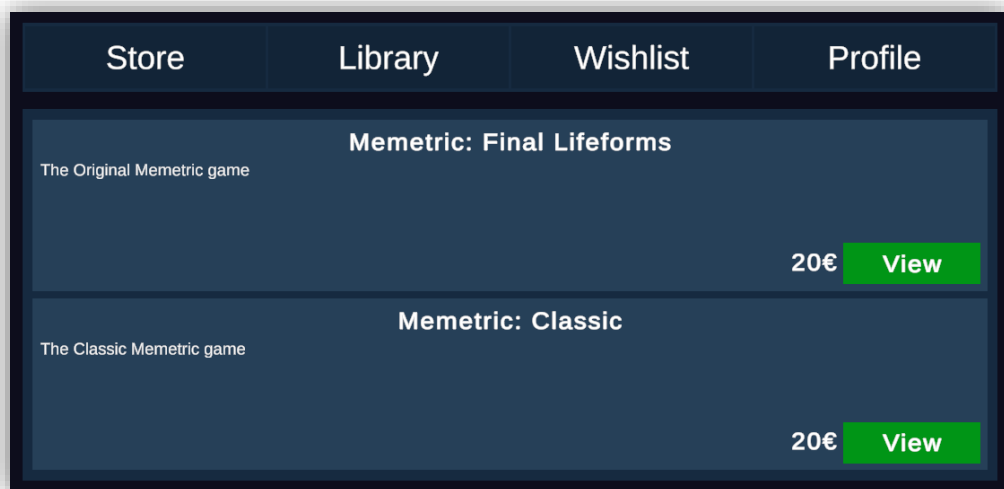
5.3.1.4 Könyvtár

A könyvtár felületen a felhasználó megtekintheti a birtokolt játékeit, valamint személyes adatokat róluk, mint hogy mennyit játszott velük, valamint a kedvencek közé tartozik-e, melyet a bal felső sarokban található csillag jelez. A játékok ABC sorrend szerint vannak rendezve úgy, hogy a kedvencek mindig a lista elején jelenjenek meg. Ezen oldal tetején szintén megtalálható a navigációs sáv.



5.3.1.5 Kívánságlista

A kívánságlista felület szinte teljesen megegyezik a bolt felülettel, azt leszámítva, hogy itt csak a felhasználó által korábban kiválasztott játékok jelennek meg.



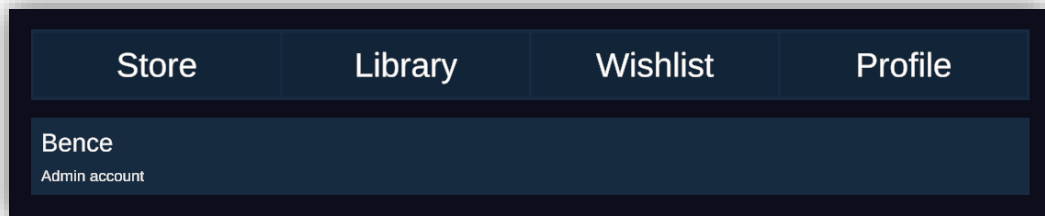
5.3.1.6 Játék

A játék felületen, mely a bolt, könyvtár, és kívánságlista felületeken át érhető el, a felhasználó részletes adatokat találhat a játékokról, mint az azok által igényelt specifikáció. Amennyiben a felhasználó nem rendelkezik a játékkal, úgy látható egy gomb az ár mellett, mely átirányít a web felületre vásárláshoz, máskülönben egy felirat tudatja a felhasználóval, hogy már birtokában van a játék. A navigációs sáv ezen felület tetején is megtalálható.



5.3.1.7 Profil

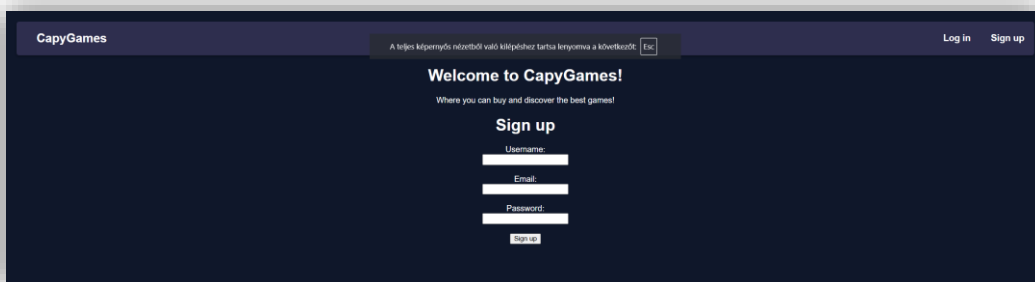
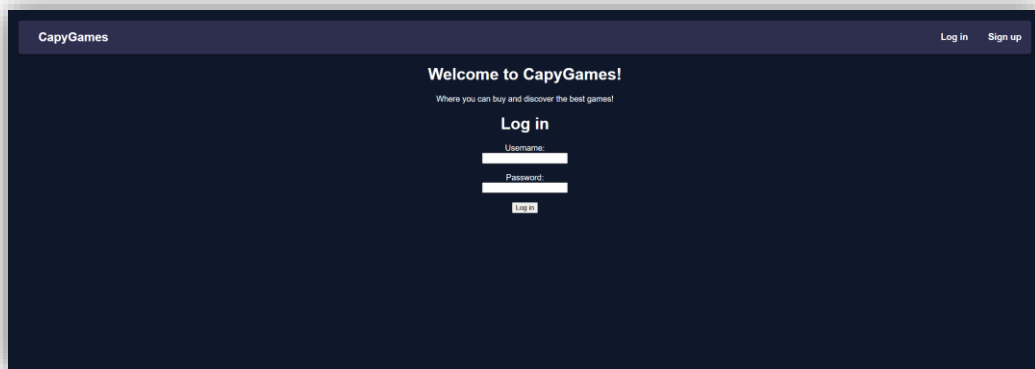
A profil felületen a felhasználó megtekintheti a saját profilját. A felület tetején megtalálható a navigációs sáv, mellyel a felhasználó más felületekre válthat.



5.3.2 Weblap felületei

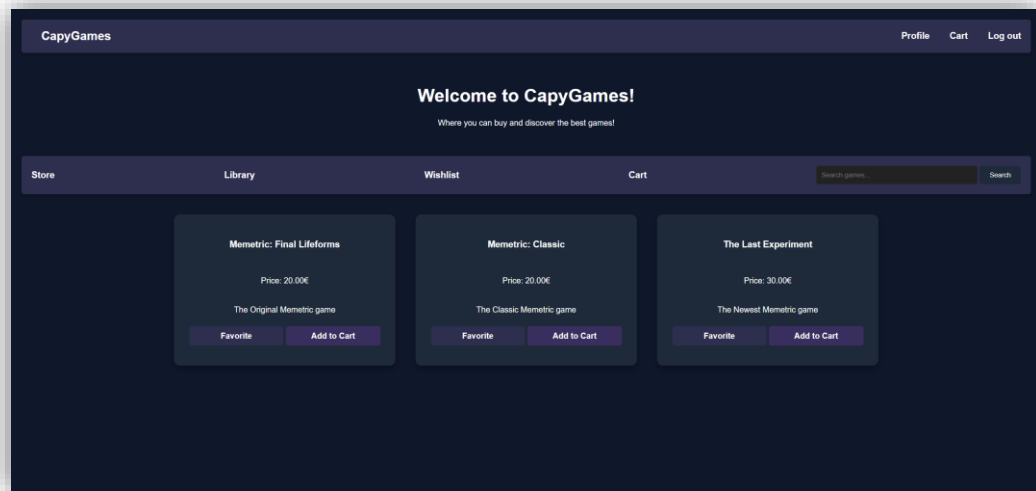
5.3.2.1 Bejelentkezés / Regisztráció

Ezen az oldalon a felhasználó bejelentkezhet vagy regisztrálhat.



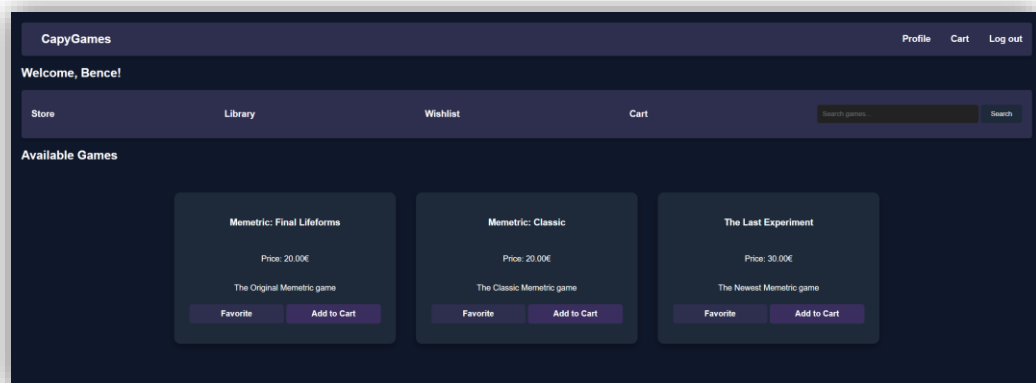
5.3.2.2 Főoldal

A főoldalon fogadja a felhasználót bejelentkezés után.



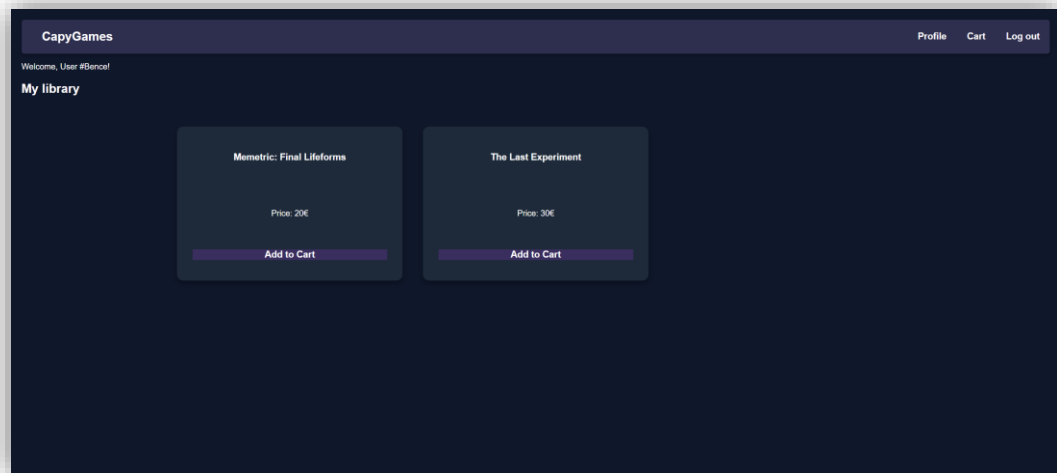
5.3.2.3 Bolt

A boltban a felhasználó böngészheti a játékokat, valamint kereset is közöttük.



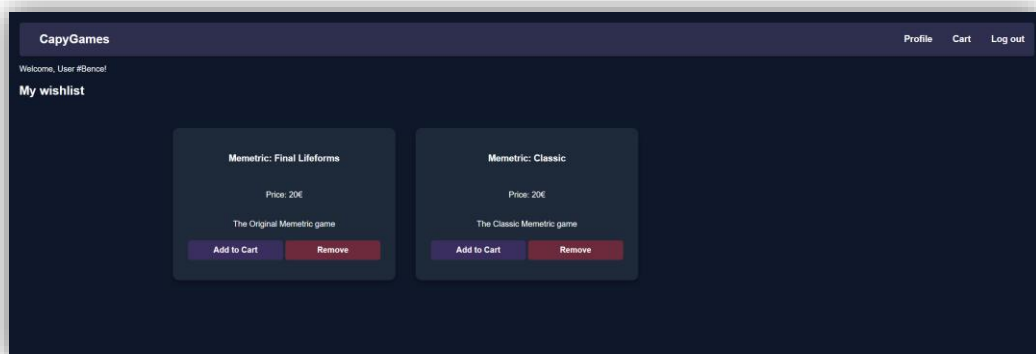
5.3.2.4 Könyvtár

A könyvtárban a felhasználó megtekintheti a birtokolt játékeit.



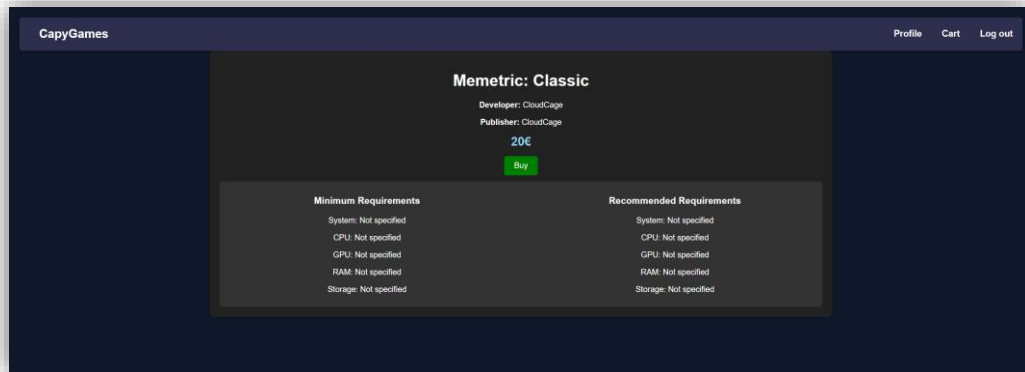
5.3.2.5 Kívánságlista

A kívánságlistán a felhasználó megtekintheti és szerkesztheti a kívánságlistáját.



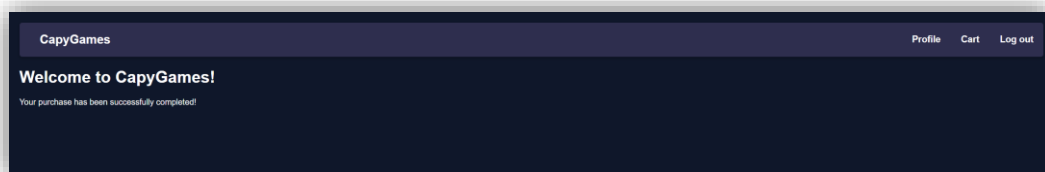
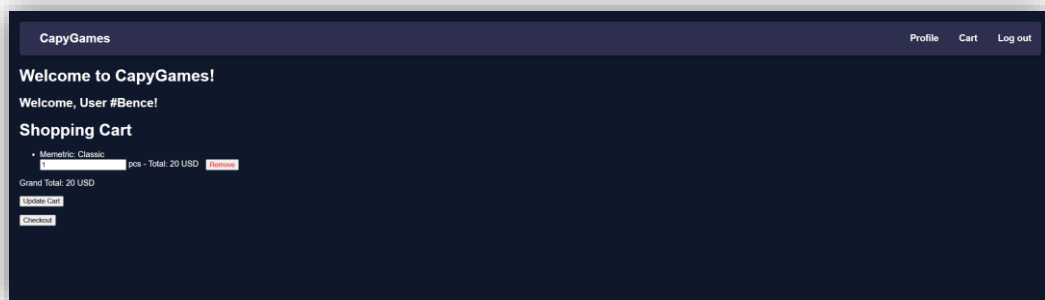
5.3.2.6 Játék

A játék nézetben a felhasználó megtekintheti egy adott játék adatait, mint például az árát, a fejlesztőt, valamint a specifikációját.



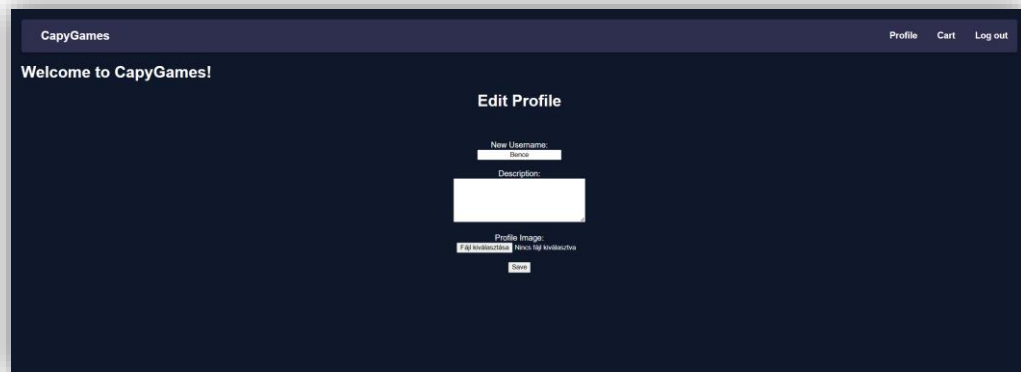
5.3.2.7 Kosár

A kosár nézetben a felhasználó megtekintheti és szerkeszteni a megvásárolni kívánt játékok listáját, valamint véglegesítheti a vásárlást.



5.3.2.8 Profil

A profil oldalon a felhasználó megtekintheti és szerkesztheti a profilját.



6 Konklúziók

Bár a projekt sok nehézséget okozott, és többször is felmerültek váratlan akadályok, komplikációk, ezeket sikerrel vettük, és elkészítettünk egy alkalmazást, mely, habár jócskán eltér az eredetileg tervezettől, egy használható alkalmazás, amiből további fejlesztések után lehetne valós termék is. Egyik ilyen akadály volt a csapattársak távozása. 3 fős csapatként kezdtük a munkát, ám 1 csapattag 2 hét után kiszállt, egy másik pedig a végén, melyet már hónapokkal előre kitalált, és nem közölte, így csak a végén értettem meg, miért dolgozott ilyen keveset a projekten, valamint hirtelen hárult rám a projekt befejezésének a feladata. Habár az alkalmazás jelenleg még nem versenyképes a nagy webáruházakkal, mint a Steam, megvan benne a lehetőség, hogy egy ütőképes versenytárssá nője ki magát, amennyiben jelen hiányosságai, fejlesztési kompromisszumai kiküszöbölésre kerülnek. Amennyiben több időm lett volna, ezeket kijavítottam volna és az eredeti specifikációra fejlesztettem volna a projektet. Összességében úgy értékelem, hogy a körülményekhez képest egy jó alkalmazást hoztunk létre.

7 Irodalomjegyzék

7.1 Online források

- <https://www.w3schools.com/>
- <https://docs.unity3d.com/Manual/index.html>
- <https://learn.microsoft.com/en-us/dotnet/csharp/>
- <https://odininspector.com/attributes>

7.2 Online eszközök

- <https://cloud.capyland.dev/> (Saját online felhő, főleg fájlmegosztásra)
- <https://github.com/>
- <https://www.apachefriends.org/> (XAMPP, a szerverkörnyezet telepítéséhez)

8 Mellékletek

- Weboldal forráskód: <https://github.com/Vizsgaremek-Projektmunka/Web>
- Rest API forráskód: <https://github.com/Vizsgaremek-Projektmunka/API>
- Asztali alkalmazás forráskód: <https://github.com/Vizsgaremek-Projektmunka/DesktopApp>
- Egyéb fájlok: <https://github.com/Vizsgaremek-Projektmunka/Others>