

2023

Game2Racker

GAME2RACKER

Search games, DLC, mods and users

KAKRANAF



New Releases

Bless



Xenonauts 2



Occupy Mars: T...



8-Bit Adventures 2



The Outlast Trials



Hogwarts Legacy



All time Top games

Gone Home



Stardew Valley



Titanfall 2



Prey



Assetto Corsa C...



Subnautica



Gyűjtsd össze a játékaidat!

Czibulya Márk, Kovács Máté,
Széklenár Andos Milán

Nógrád Megyei Szakképzési Centrum
Szent-Györgyi Albert Technikum

szakma megnevezése:

szoftverfejlesztő és -tesztelő technikus

szakma azonosító: 5-0613-12-03

2023.04.26

Tartalom

Bevezetés	3
1. Felhasználói dokumentáció.....	5
1.1. Főoldal.....	5
1.2. A játék bemutató oldala	7
1.3. Keresés	8
1.4. Profil.....	9
1.5. Beállítások.....	11
1.6. Regisztráció.....	11
1.7. Bejelentkezés.....	12
1.8. Lábléc (footer).....	12
1.9. Comment szekció	13
2. Felhasznált technológiák.....	15
2.1. HTML	15
2.2. CSS.....	15
2.3. JavaScript	16
2.4. PHP	16
2.5. Visual Studio Code	16
2.6. GitHub.....	17
2.7. XAMPP	17
2.8. API	18
3. Adatbázis dokumentáció.....	19
3.1. Az adatbázis célja.....	19
3.2. IGDB API.....	19
3.3. Szükséges információk.....	21
3.4. Kapcsolatok meghatározása	21
3.5. Users tábla.....	22

3.6.	Sessions tábla	23
3.7.	ratingTable	23
3.8.	Comment tábla	24
3.9.	Ratios tábla.....	25
3.10.	lists tábla.....	26
3.11.	listGames tábla	26
3.12.	games tábla	27
3.13.	Adatbázis diagram	28
4.	Fejlesztői dokumentáció	29
4.1.	Globális fájlok.....	29
4.1.1.	tokenHandler.php.....	29
4.1.2.	navbar.php.....	31
4.1.3.	curl.php	31
4.1.4.	comments.php	32
4.1.5.	rating.php	33
4.1.6.	lists.php	34
4.2.	Regisztráció.....	38
4.3.	Bejelentkezés.....	40
4.4.	Játék oldal.....	42
4.5.	Felmerült problémák a fejlesztés során.....	44
	Összefoglalás	44
	Források	46
	Ábrajegyzék	47

Bevezetés

Csapatunk 3 főből áll, Czibulya Márk, Kovács Máté és Szklenár Andos Milán. A témaválasztás kapcsán több ötletünk volt, ezek közül volt olyan is, amit el is kezdtünk megvalósítani, de különböző fejlesztési és feladat-megosztási akadályokba ütköztünk.

Az első elképzelésünk a COVID-19 vírus okozta iskolaszünet kapcsán született meg, ugyanis a lezárás időszakában nem volt lehetőségünk a mindennapos iskolába járásra fél éven keresztül, ezért otthonról online oktatás formájában tudtunk részt venni a tanórákon. A bezártság kapcsán jutott az eszünkbe egy olyan szolgáltatás vagy koncepció, hogy együtt tudjunk videókat nézni az interneten. Így megálmodtuk a Gather2Watch-t.

Ilyen alkalmazás már létezik a világhálón, de kutakodva arra a döntésre jutottunk, hogy mi szeretnénk ezt egy továbbfejlesztett formában létrehozni, illetve ingyenessé tenni, mert jelenleg csak fizetős alkalmazás létezik az interneten.

Ezt úgy terveztük meg, hogy a weboldalunkra történő autentikációt követően a belépett felhasználók közösen, egy adott videót egyszerre tudják nézni a társaikkal. Ehhez nem kell külön-külön megnyitniuk egy videó-megosztó weboldalt, hanem csak egy linkre van szükségük a videó megtekintéséhez. Ugyanakkor a regisztrált felhasználók a beépített chat felületen meg tudták volna beszélni a látottakat.

Ennek megvalósításához olyan websocket megoldást találtunk, amihez Nodejs backend technológiát kellett volna használnunk. El is kezdtük vele az ismerkedést, és a regisztrációt meg is tudtuk valósítani, valamint a chat is kiválóan működött, azonban itt elakadtunk. Egyrészt technológiai nehézségek miatt, mivel backend tantárgyból a PHP-t tanultuk és nem a Nodejs-t. Másrészt a feladatokat sem tudtunk igazán egyenlően elosztani egymás között, hogy az mindannyiunk számára értékelhető projekt legyen.

Ezt követően gondolkoztunk tovább, hogy valamilyen focial kapcsolatos alkalmazást készítsünk, de ez a téma nem érdekelte a csapatunk mindent tagját, így ez már az ötletelés fázisában elvetésre került.

Végül kézenfekvő volt a következő ötlet, ami mindenkit érdekel, és ez a számítógépes játékok. Nem egy konkrét játék fejlesztését terveztük, hanem szintén egy közösségi oldalon gondolkoztunk, ahová azok regisztrálhatnak, akik érdeklődnek a számítógépes játékok iránt, az általunk létrehozott felületen megoszthatják a tapasztalataikat, és információkat szerezhetnek a játékokról, szavazhatnak a különböző játékokra 1-től 10-ig terjedő pontozással.

A különböző feladatokat is fel tudtuk osztani egymás között. A csapatunkból Máté készítette el a regisztrációs, belépési felületet és a minden oldalon megjelenő „footert” ami olyan hasznos lehetőségeket kínál. Márk az Internet Game Database oldalához kapcsolódó lekérdezéseket írta meg. Milán egyrészt a főoldalon megjelenő tartalmat fűzte össze, valamint kidolgozta a főoldalon megjelenő következő funkciókat: ha a látogató az egyes játékok kártyáira kattint, akkor onnan a kiválasztott játék egyedi oldalára jutva további információkat tekinthet meg róla, illetve igénybe veheti a hozzászólási lehetőségeket.

Fontos megjegyeznünk, hogy az alkalmazásunkat eleve angol nyelven készítettük el, mivel nem szeretnénk, ha a felhasználók köre egy szűk földrajzi környezetre korlátozódna. Jövőben célunk, hogy egy nemzetközi célcsoportot érjünk el, és a világ minden tájáról regisztráljanak. Ezt az angol nyelv gyakorlása, és nemzetközi kapcsolatok építése miatt is fontosnak tartjuk.

A megvalósításhoz a PHP, JavaScript, MySQL technológiákat használtuk. Az adatok tárolása adatbázisban történt, melyből PHP segítségével nyerjük ki az adatokat, ezáltal egy nagyon könnyen karbantartható weboldalt készítettünk.

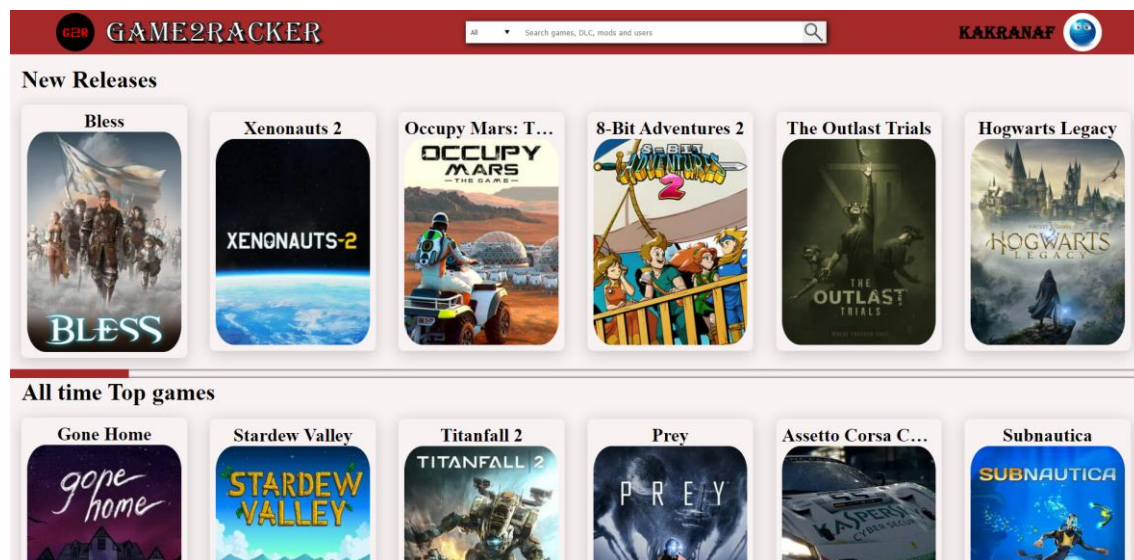
Dolgozatunk első részében bemutatjuk a fentebb említett, felhasznált technológiákat, és a fejlesztői környezetet, ezt követően a felhasználói dokumentációban az oldal részletes leírását, a fejlesztői dokumentációban az oldal frontend és backend oldali programozását.

Külön fejezetbe gyűjtöttük össze a programozás során felmerült akadályokat, valamint azt, hogyan oldottuk meg ezeket. Végezetül az összefoglalásban értékeljük a közös munkánkat és ismertetjük azokat az ötleteinket, amelyek már nem kerültek megvalósításra, de úgy gondoljuk, hogy majd a jövőben mindenképpen érdemes lenne megvalósítanunk ezeket a továbbfejlesztési lehetőségeket.

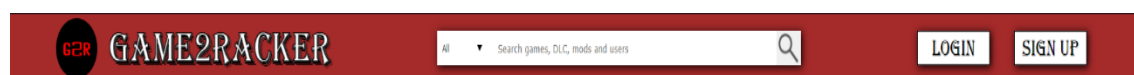
1. Felhasználói dokumentáció

A weboldalon számítógépes játékok gyűjteménye jelenik meg. A felhasználók regisztrálhatnak a weboldalra, értékelhetik a játékokat és a felhasználókat is a profiljuk és a hozzászólásaik alapján.

1.1. Főoldal



A Game2Racker (Angolul szokták a „to” szót rövidíteni 2-vel. A „to racker” szófordulat hasonlít a „Tracker” szóra ami nyomon követést jelent) weboldalt megnyitva a főoldalon találjuk magunkat. A navigációs sáv közepén található egy két részre osztott keresőmező. Az első részben egy lenyitható választómenü látható, ahol választhatunk a játékok, a DLC¹-k, a felhasználók által írt mod²ok, illetve az oldalon regisztrált felhasználók között. Ha kiválasztjuk például a felhasználókat és végrehajtunk egy



1. ábra Navigációs sáv

keresést, akkor a felhasználók profiljai jelennek meg az oldalon.

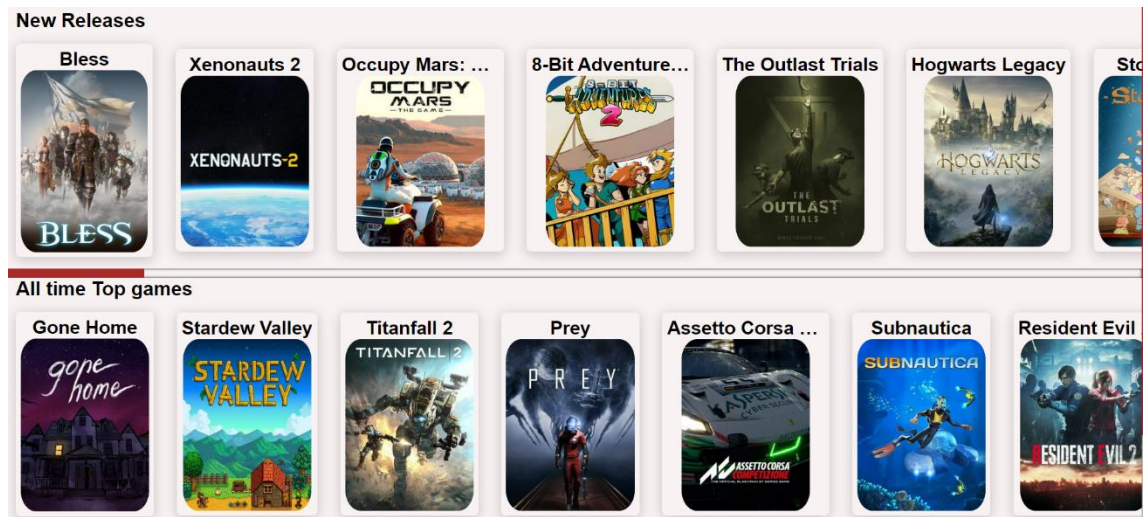
A jobb felső sarokban található meg a belépés és a regisztráció gomb. Bejelentkezés után a jobb felső sarokban megjelenik a felhasználó neve, amelyre rákattintva megjelenik

¹ A DLC már megjelent játékokhoz, az alapjáték fejlesztője által készített kiegészítő. A kiegészítőket az interneten keresztül osztja meg a játék készítője.

² A videojáték-módosítás az a folyamat, amikor a játékosok vagy rajongók megváltoztatják a videojáték egy vagy több aspektusát, például azt, hogy hogyan néz ki, vagy hogyan viselkedik, és ez az általános modding egy részterülete.

egy lenyíló menü, melynek segítségével megtekinthetjük a saját profilunkat, vagy a beállításainkat, és ki is tudunk jelentkezni a fiókunkból.

A főoldalon található meg a legújabb megjelenésű játékok listája, illetve a legtöbbet játszott játékok a felhasználók értékelései alapján. Egy csúszka segítségével könnyen tudunk böngészni a játékok között.

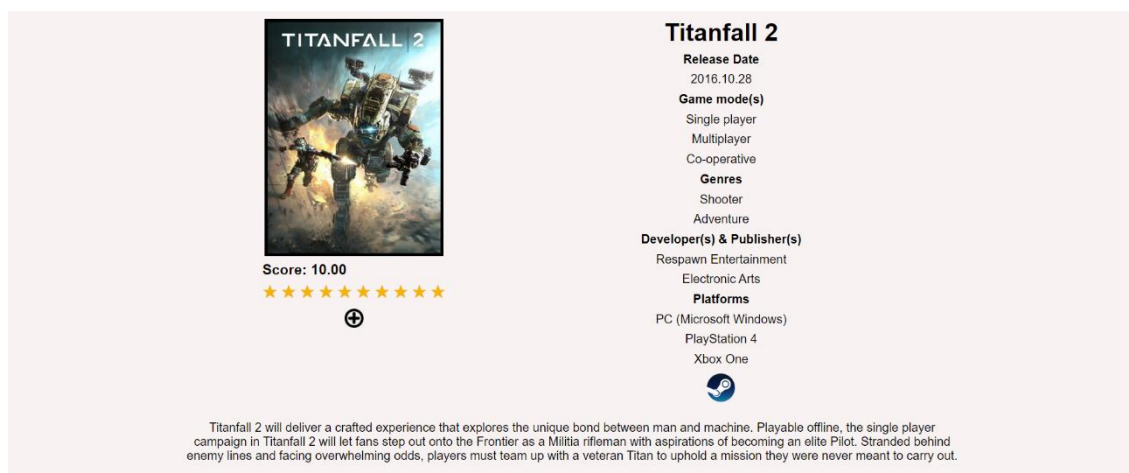


2. ábra Főoldalon megjelenő játékok

Megvalósításra került még egy lábléc is a főoldalon, amelyre több hasznos funkció került. Itt vissza tudunk navigálni a főoldalra a „Home” gombra kattintva, valamint található még egy „Profile” gomb is, mely a saját profiljára navigálja a felhasználót, ha be van jelentkezve. Szintén megtalálható még egy úgynevezett ÁSZF, amely a chat funkció szabályait tartalmazza, valamint sok más hasznos dolog is, melyekre később részletesebben kitérünk.

1.2. A játék bemutató oldala

Ha a főoldalon, vagy a keresésből rákattintunk egy játékra, megjelenik egy új oldal, ahol a játékról minden lényeges információt megtalálunk. Megjelenik a játék borítóképe, a felhasználók értékelése, a listához adáshoz szükséges gomb, a játéknak a neve, megjelenési dátuma, játékmódja, kategóriája, fejlesztők és kiadók nevei, platformok melyekre megjelent az adott játék, webhelyek mint például Steam vagy Epic Games, ahol könnyedén megvásárolható a játék. Található még itt egy rövid leírás a játékokról, egy videó, illetve képek a játékról és a felhasználóink által írt hozzászólások.



Titanfall 2

Release Date
2016.10.28

Game mode(s)
Single player
Multiplayer
Co-operative

Genres
Shooter
Adventure

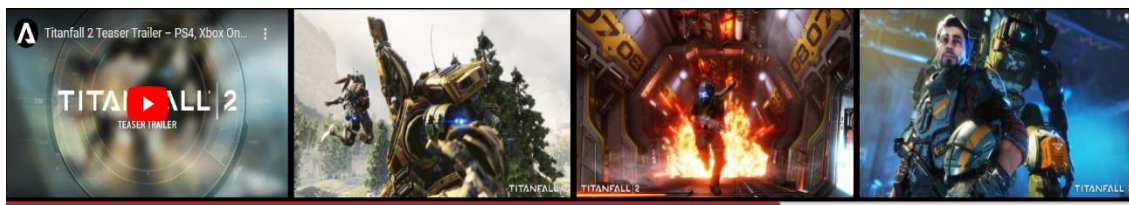
Developer(s) & Publisher(s)
Respawn Entertainment
Electronic Arts

Platforms
PC (Microsoft Windows)
PlayStation 4
Xbox One

Score: 10.00
★★★★★★★★★

Titanfall 2 will deliver a crafted experience that explores the unique bond between man and machine. Playable offline, the single player campaign in Titanfall 2 will let fans step out onto the Frontier as a Militia rifleman with aspirations of becoming an elite Pilot. Stranded behind enemy lines and facing overwhelming odds, players must team up with a veteran Titan to uphold a mission they were never meant to carry out.

4. ábra Egy játék adatai

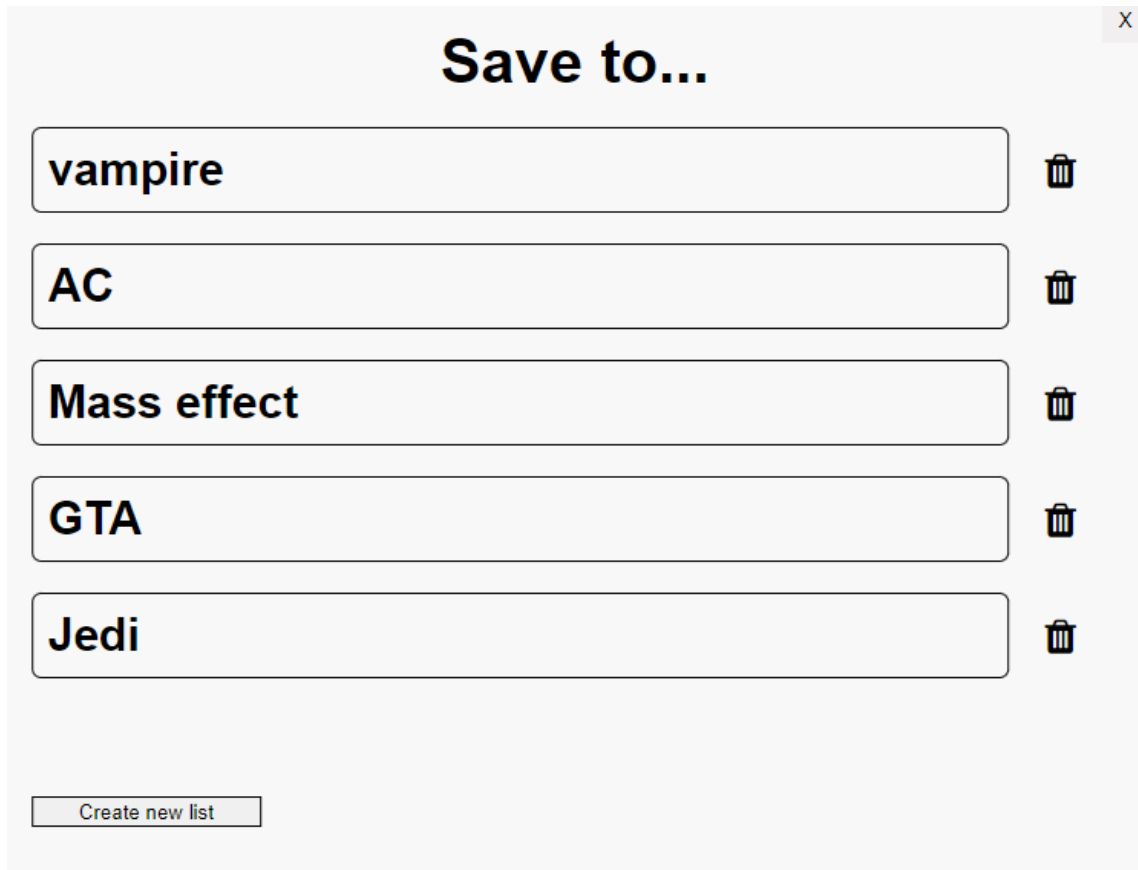


3. ábra Videó és képek a játékról

Ha a felhasználó szeretne értékelni egy játékot, akkor ahhoz be kell jelentkeznie, ellenkező esetben a csillagok nem kattinthatók. Ezután egy 1-től 10-ig terjedő skálán értékelheti a játékot. A Score³ funkció a következőképpen működik: a felhasználók értékelései alapján számol egy átlagot, és így jelenik meg egy kapott szám a weboldalon a Score mellett.

³ Pontszám

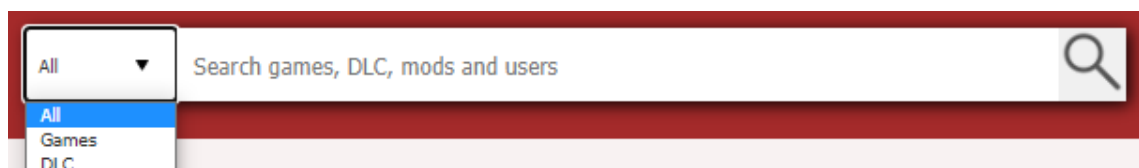
A plusz gombbal a felhasználó egy saját maga által létrehozott listába tudja rakni a játékot, melyet egy felugró ablakban tehet meg. A felugró ablakban lehetősége van a felhasználónak új listát is létrehozni, nevet adni a listának, a játékokat elmenteni a kiválasztott listába és törölni a létrehozott listáját.



5. ábra Listák

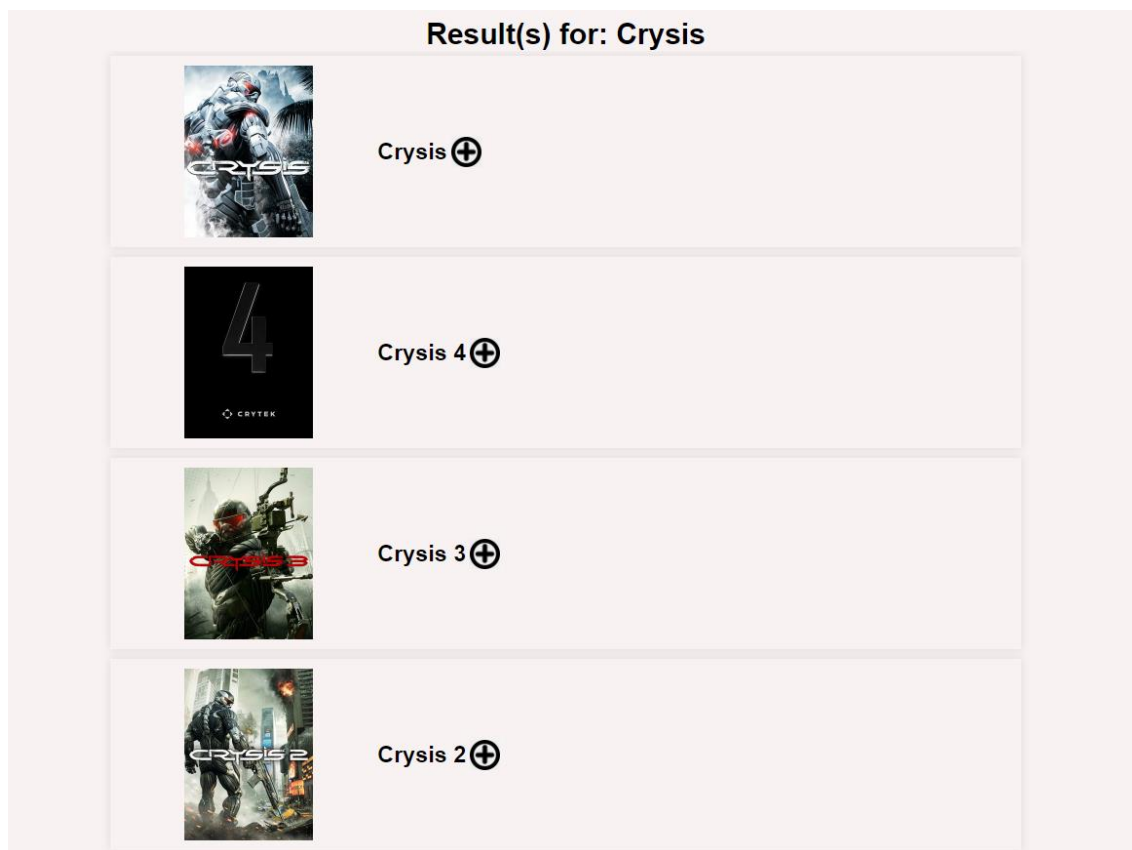
1.3. Keresés

A navigációs menüben, amikor kiválasztotta a felhasználó, hogy milyen keresést szeretne végrehajtani (All, Games, DLC, mods) majd beírja a keresőbe a játék nevét, vagy annak egy részletét, egy új oldalon egymás alatti kártyákban megjelenik a játékok képe és neve.



6. ábra Keresés

Ezek a kártyák kattinthatók és átirányítanak az adott játék bemutató oldalára. A játék neve mellett még a listához adás is megtalálható, ha a felhasználó ismeri a játékot és egyből listához szeretné adni.



7. ábra Keresési eredmények

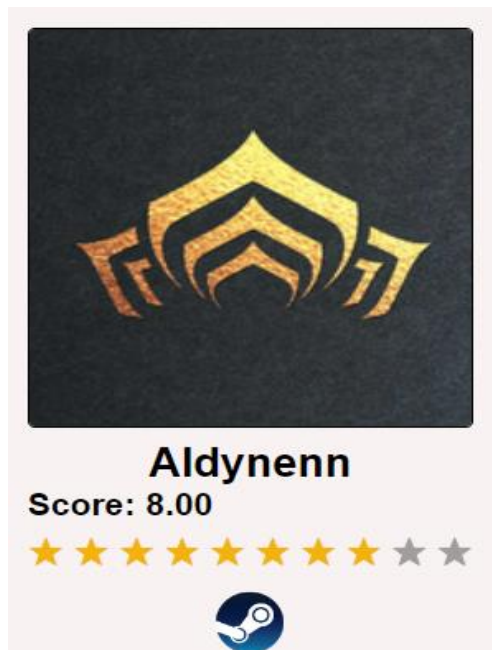
1.4. Profil

A navigációs menüben a felhasználónak lehetősége van megtekinteni saját profilját a profilképére kattintva, vagy a nevére kattintva lefelé nyíló menüben a „View Profile”-ra lépve.



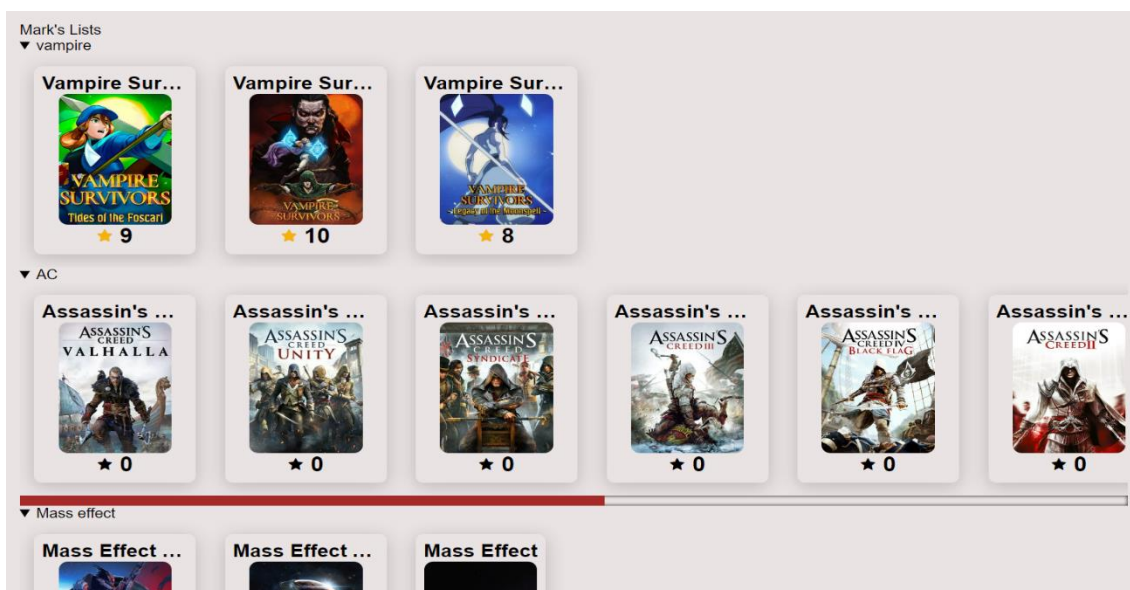
8. ábra Profil menü

A profil oldalon látható a felhasználó profilképe, a Score-ja amit más felhasználók adtak neki. A saját profilját, nem tudja értékelni a felhasználó, de más profilját igen, amit ugyanúgy egy 1-től 10-ig terjedő skálán tud megtenni és átlagolja a Score-t, mint ahogy a játékoknál.



9. ábra Profil kép és score

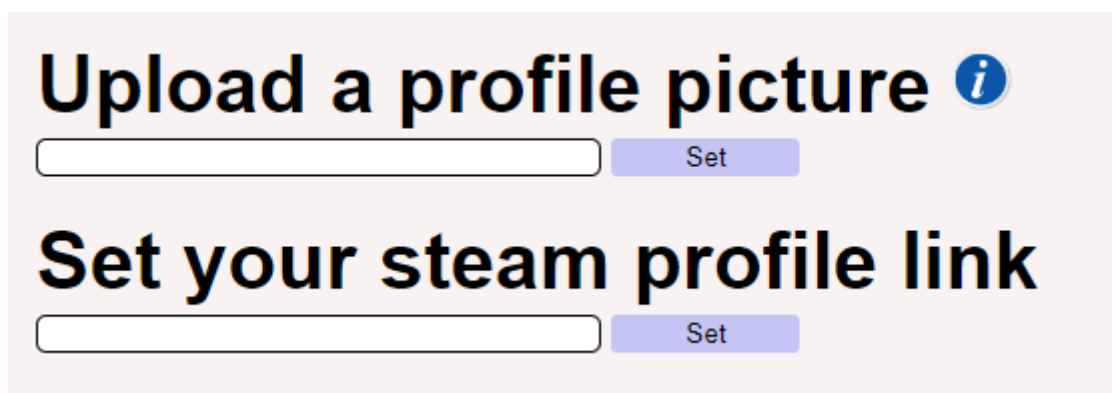
A profilkép mellett megjelennek a felhasználó által létrehozott listák is, ahol egy csúszkán látja az összes játékot, amit a listájához adott. A játékok alatt pedig látható a felhasználó értékelése a játékról. A profiloldal alján megjelenik a comment szekció is.



11. ábra Felhasználó listái

1.5. Beállítások

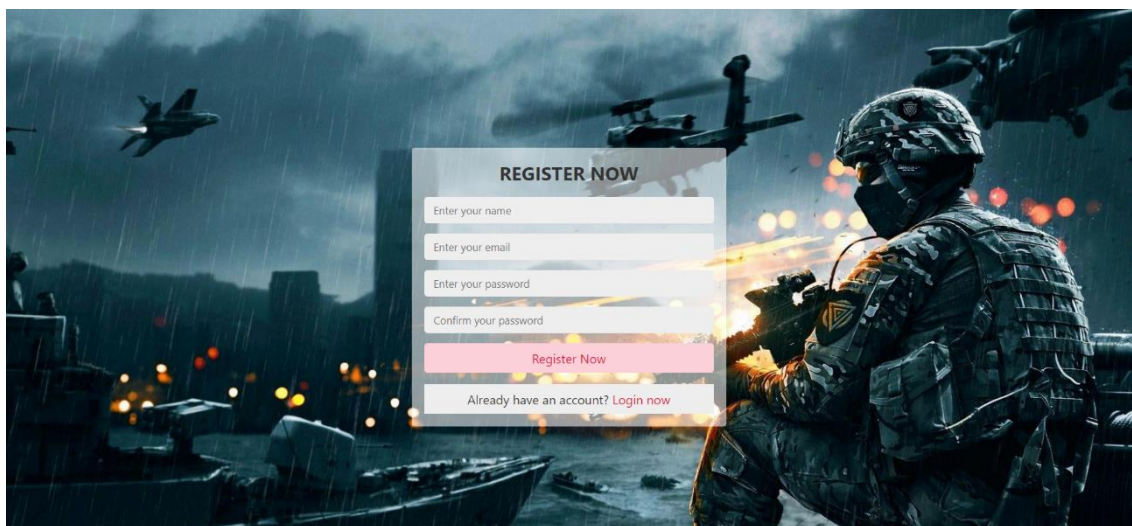
A navigációs menüben a nevünkre kattintva lenyílik egy almenü, ahol a fogaskerékre kattintva átjutunk a beállítások oldalra. Itt a felhasználó be tud állítani egy profilképet egy linken keresztül (képeket nem tárolunk az adatbázisban, csak a hozzáférési útját), illetve a Steam⁴ profiljának a linkjét, ami megjelenik a profilján egy kattintható Steam ikonként.

The image shows a user interface for profile settings. At the top, there's a heading "Upload a profile picture" followed by an information icon. Below it is a text input field and a "Set" button. Further down, there's a heading "Set your steam profile link", another text input field, and a "Set" button. The background is a light pinkish-grey.

12. ábra Beállítások

1.6. Regisztráció

A regisztráció gombra kattintva, megjelenik egy regisztrációs felület, ahol a felhasználónak meg kell adnia egy általa választott egyedi felhasználónevet, az e-mail címét, illetve a jelszavát. Létrehoztunk egy jelszó megerősítő mezőt is, hogy a regisztráló

The image shows a registration form overlaid on a game background. The form has the title "REGISTER NOW" and four input fields: "Enter your name", "Enter your email", "Enter your password", and "Confirm your password". Below the fields is a red "Register Now" button and a link that says "Already have an account? Login now". The background is a dark, rainy scene from a game, featuring a soldier in the foreground and various vehicles in the background.

13. ábra Regisztráció

⁴ A Steam egy tartalomtovábbító és -kezelő rendszer, amelyet a Valve Software fejlesztett ki. Funkciói különféle számítógépes szoftverek digitális áruházi rendszerben történő értékesítése, többjátékos módok menedzselése, és közösségépítő háló fenntartása.

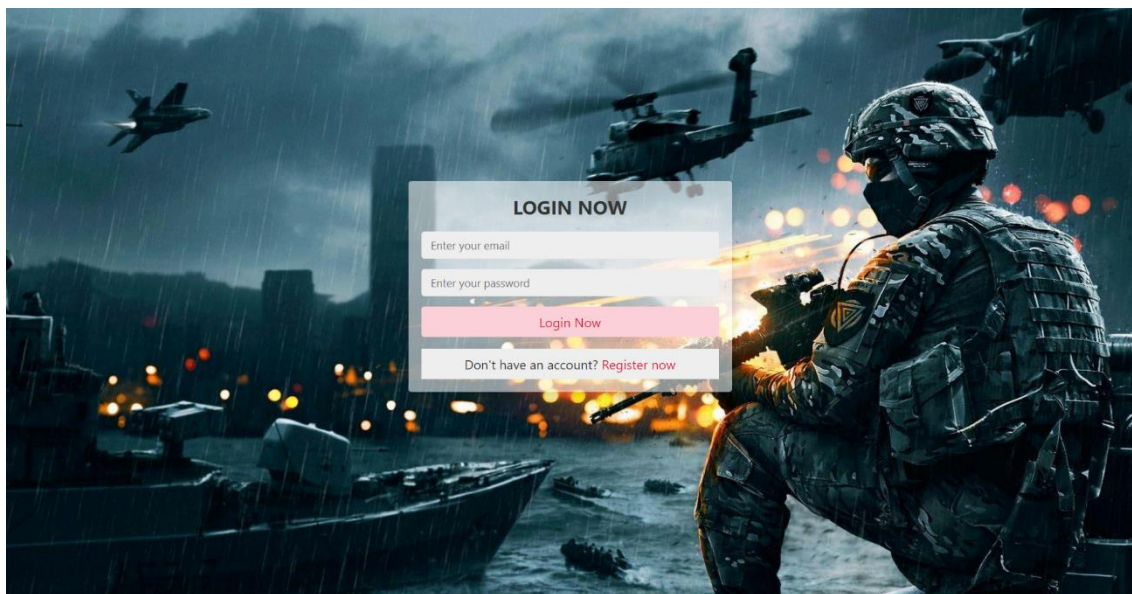
meggyőződhesen arról, hogy a jelszavát jól írta-e be, ezáltal elkerülve a későbbi esetleges fennakadásokat.

A regisztrációt nagyon fontosnak tartjuk, mivel nagyon sok funkció csak regisztrálás után érhető el, ennek érdekében több helyen is feltüntettük a regisztráció gombot az oldalunkon.

1.7. Bejelentkezés

A bejelentkezés gombra kattintva megjelenik a belépési felület, ahol be tudunk lépni a profilunkba előzetes regisztrációt követően. A felhasználónak a belépéshez meg kell adnia az e-mail címét illetve a regisztráció során megadott jelszavát. Itt egy olyan lehetősége is van a látogatónak, ha például még nem rendelkezik regisztrációval az oldalunkon, akkor „Register Now” gombra kattintva megteheti ezt.

Sikeres bejelentkezés után a főoldalon találjuk magunkat. Ha a belépés sikertelen, akkor hibaüzenetet kapunk: „Incorrect password or e-mail”. Ilyenkor lehetőségünk van újból megpróbálni bejelentkezni az oldalunkra.



14. ábra Bejelentkezés

1.8. Lábléc (footer)

Létrehozásra került a weboldal alján egy lábléc, mely nagyon hasznos információkat rejt a felhasználók számára. Itt lehetőségünk van olyan funkciók használatára, amit a felhasználó esetleg feljebb nem vett észre, mint például a bejelentkezés, regisztráció vagy éppen a kijelentkezés. Ezeket pontosan azért raktuk ide,

hogy ha a felhasználó elfelejt regisztrálni vagy éppen belépni, akkor ezt még egy helyen könnyedén megtalálja. Ezért tartottuk ideálisnak a láblécbe elhelyezni ezeket a hasznos funkciókat.

Létrehozásra került még egy Home gomb, mely visszavigyél minket a főoldalra, egy Profile gomb, amelyre kattintva eljuthatunk a profilunkra könnyedén, illetve egy General Terms and Conditions, amely az általános szerződési feltételekre navigál minket, ahol megtekinthetjük a Game2Racker oldal legfontosabb szabályait.

Game2Racker General Terms and Conditions

§ 1 General information

By registering for the game, the user is requesting to enter into a free user agreement with Game2Racker. In addition, by providing his email address, the user states his consent to the verification of his email address (see § 2). The operator accepts this request by granting a user account that provides access to the game. The user agreement is based on these general terms and conditions, which the user acknowledges by registering. There are no subsidiary agreements. The operator enters into user agreements exclusively with natural persons. The user agreement is valid for an indefinite time period and can be canceled in writing, generally by email, at any time by both parties without adhering to a particular schedule. The user does not have any claim on opening an account, being permitted to participate in the game, the maintenance of the game or its accessibility. The operator reserves the right to continue developing the game or to change it at any time or cancel its operation without stating a reason. The operator may change these general terms and conditions at any time. The operator will notify the user of any changes explicitly and in a timely fashion. By playing the game again (logging in) after the notification has been sent, the user declares his acceptance of the changed general terms and conditions.

§ 2 Account / ownership of virtual goods

The name can contain letters, numbers, spaces and some special characters, but it cannot consist only of numbers. When choosing an account name, the user must respect the relevant privacy rights of living/existing persons. The same applies similarly to the names of groups, which accordingly cannot be named after existing organizations, companies or brands. All accounts, their values, resources, objects, etc. virtual goods in the game. The user obtains exclusive right to use his user account and non-exclusive right to use all other virtual goods. The usage rights are limited in time depending on the validity of the user agreement, unless a shorter time is specified in the game. The user does not have any property rights or other rights regarding the virtual goods beyond the right of use.

§ 3 Exploiting bugs & scripting / cheating

The user may not perform actions that burden the server with an excessive amount of data transfer (for example, automatic updating). Automatic or semi-automatic scripts that query the database or start game mechanisms are also strictly prohibited. In case of this type of violation of the Terms and Conditions, the corresponding account will be blocked or deleted without prior warning. The operator reserves his claim for compensation for the damage caused and the costs incurred in this regard, and his right to initiate the procedure to enforce this. All players are obliged to immediately report any program errors they recognize to the operator. Anyone who uses the above program errors to gain an advantage for himself or others may expect to have his account blocked. Deliberately causing any program error is only permitted if the player concerned is called to do so by a customer service employee.

15. ábra ÁSZF

A harmadik menüpontban megismerkedhetünk a közösségi oldalainkkal, ahol könnyedén felvehetjük a kapcsolatot az adminokkal, illetve hasznos posztokat, híreket is találhatunk a közösségi médiánkban. Facebook, Twitter és Instagram közösségi oldalakon leszünk megtalálhatók.




Quick Access

- Login
- Register
- Logout

Navigation

- Home
- Profile
- General Terms And Conditions

Follow Us



16. ábra Lábléc (footer)

1.9. Comment szekció

A Comment szekcióban hozzászólást írhat a belépett felhasználó (legyen az oldal egy adott felhasználó profilja, vagy egy játék oldala).

A like és dislike gombbal reagálhat minden felhasználó egy hozzászólásra, ez az összes interakciót számolja, ezért lehet mínusz értékelése is egy hozzászólásnak.

Megjelenik még a felhasználó avatarja és felhasználóneve is. A felhasználónévre kattintva átirányít a profiljára.

Mellette látható az is, hogy mikor írta a felhasználó a hozzászólást. Egy felhasználó a saját hozzászólását tudja törölni, ha elgépelte valamit vagy csak szimplán meggondolta magát. Ha a felhasználó a saját profilján van, akkor mindenki hozzászólását tudja törölni.

Comments

Write a comment...

Comment

-1

**Mate**

Most relaxing game to ever exist.

2023-04-24 08:28:40

1

**Mark**

One of the best game ever!

2023-04-24 08:24:22

17. ábra Hozzászólások

2. Felhasznált technológiák

2.1. HTML

HTML (HyperText Markup Language) egy szöveges leírónyelv, amelyet az interneten megjelenő oldalak létrehozásához használnak. Magyarul a "hipertext jelölőnyelv" kifejezést használhatjuk rá. Az HTML segítségével lehetővé válik az oldal szerkezetének és formázásának megadása, valamint különböző interaktív elemek (pl. hivatkozások, képek, videók, űrlapok) beillesztése az oldalra.

A HTML-t egy szerver oldali jelölő nyelvként használják, ami azt jelenti, hogy az oldalt kérő kliens a szerverről kapja meg az HTML forráskódját, amit a böngésző megjelenít a felhasználó számára. Az HTML forráskódja szöveges állományokban tárolódik, és a böngészők a kód alapján jelenítik meg az oldalt a felhasználó számára. A forráskódban az HTML elemeket címkék (tag-ek) között írják le, és az elemek tulajdonságait attribútumok segítségével állítják be.

Az HTML-t általában CSS (Cascading Style Sheets) segítségével formázzuk, ami lehetővé teszi az oldal megjelenésének (pl. színek, betűtípusok, elrendezés) testreszabását. Ezen kívül az HTML-t JavaScript segítségével is lehet interaktívvá tenni, ami lehetővé teszi az oldalakon előforduló események kezelését és az oldal dinamikus működését.

2.2. CSS

A CSS (Cascading Style Sheets) egy nyelv, amely lehetővé teszi a weboldalak formázásának és kinézetének szabályozását. A CSS segítségével beállítható a szöveg betűtípusa, színe, háttere, elrendezése, stb. A CSS lehetővé teszi az oldalak egységes kinézetének kialakítását, és elkülöníti a tartalmat a formázástól, ami javítja a weboldal karbantarthatóságát és a fejlesztői hatékonyságot. Egy stíluslapot több oldalhoz is hozzárendelhetünk, így rugalmasságot és időt spórolhatunk velük. Ha a böngészőben megnyitunk egy oldalt, akkor az egy úgynevezett cache (gyorsítótár) mappába lementi a stíluslapot, és ha legközelebb az oldalra navigálunk, akkor nem kell megvárni, hogy letöltődjön a CSS fájl, mert a böngésző már a gyorsítótárban eltárolta azt.

A CSS használatával egy weboldal HTML kódjában található elemekhez rendelhetők stílusok, ami nagymértékben növeli a weboldal esztétikai megjelenését.

2.3. JavaScript

A JavaScript egy szkript nyelv, ami azt jelenti, hogy fordítás nélkül, közvetlenül a böngészőben fut. A JavaScript-et általában HTML és CSS dokumentumokkal együtt használják, hogy weboldalak hozzanak életre. A HTML adja meg az oldal alapstruktúráját, míg a CSS formázást és stílust ad az oldalnak. A JavaScript pedig lehetővé teszi az oldal interaktív funkcióinak létrehozását.

A JavaScriptet a W3C (World Wide Web Consortium) fejlesztette ki és az ECMAScript (European Computer Manufacturers Association) standard alapján működik. Széleskörben használt nyelv, minden modern webes böngésző támogatja, és lehetővé teszi számos különböző eszközön való futtatását: asztali számítógépeken, mobil eszközökön, IoT eszközökön, stb.

2.4. PHP

A PHP egy szerveroldali script nyelv, amelyet weboldalak és webalkalmazások készítésére használnak. Az "PHP: Hypertext Preprocessor" rövidítése, és egy nyílt forráskódú programozási nyelv, amelyet széles körben használnak dinamikus weboldalak létrehozására. A PHP kódot közvetlenül HTML kódba lehet beágyazni, és a kiszolgáló feldolgozza, mielőtt az eredményként kapott HTML küldené el a felhasználó webböngészőjébe.

A PHP alkalmas webfejlesztésre, mert széles körű webalkalmazásokat tudunk vele készíteni, egyszerű weboldaltól a bonyolult webalapú rendszerekig. Nagy és aktív közösséggel rendelkezik, amely sok útmutatót és segítséget kínál egymásnak, és számos készen álló keretrendszert és könyvtárat tartalmaz, mint például a Laravel, CodeIgniter, Symfony és még sok más, amelyek segítenek a fejlesztőknek gyorsan és egyszerűen webalkalmazásokat készíteni.

A PHP többféle adatbázissal is kommunikálhat, beleértve a MySQL, PostgreSQL és SQLite, és kompatibilis számos különböző operációs rendszerrel, beleértve a Windows, Linux és macOS platformokat. A PHP jól együttműködik más technológiákkal, mint például a HTML, CSS és JavaScript, amelyeket általában a weboldal elsődleges részének létrehozására használnak.

2.5. Visual Studio Code

A Visual Studio Code egy ingyenes és nyílt forráskódú szoftverfejlesztői környezet, amelyet a Microsoft fejleszt. Segítségével többféle programozási nyelven írhatunk kódot,

például C#, C++, JavaScript, Python. Emellett számos hasznos eszközzel rendelkezik, mint például a kód automatikus kiegészítése, hibakeresés, a kód formázása és a verziókezelés.

Az egyik legnépszerűbb szoftverfejlesztői környezetnek számító VSCode használatával egyszerűen és hatékonyan tudjuk fejleszteni a különböző projekteket, ráadásul több platformon is elérhető Windows, Mac, és Linux rendszeren egyaránt.

2.6. GitHub

GitHub egy web alapú platform, amelyet főként verziókezelésre és együttműködő szoftverfejlesztésre használnak. Többféle funkciót biztosít a kód-tárolók kezeléséhez, mint például forráskód-kezelés és verziókezelő eszközök. A fejlesztők többnyire a GitHub-ot használják, hogy meg tudják osztani kódjukat egymással, együttműködjenek projekteken és hozzájáruljanak az open-source szoftverekhez.

A GitHub 2008-ban alakult és azóta az egyik legnagyobb kódtároló platformmá vált a világon. Egyéni fejlesztők milliói használják, illetve rengeteg szervezet, köztük néhány legnagyobb tech cég is, mint a Microsoft, a Google és a Facebook.

2.7. XAMPP

Az XAMPP egy ingyenes és nyílt forráskódú szoftvercsomag, amely tartalmazza az Apache web szervert, a MySQL adatbázis-kezelő rendszert, valamint az PHP és a Perl programozási nyelveket. A neve a csomagban található komponensekre utal: "X" az operációs rendszer (Cross-platform), "A" az Apache, "M" a MySQL, "P" pedig a PHP rövidítése.

Az XAMPP célja, hogy egyszerűen telepíthető és használható környezetet biztosítson a webfejlesztéshez. Általában lokális számítógépeken használják, hogy teszteljék és fejlesszék a weboldalakat, mielőtt azokat az internetre feltöltik.

Az XAMPP telepítése és konfigurálása viszonylag egyszerű, sok dokumentáció és támogatás áll rendelkezésre az interneten. Az XAMPP használatával a webfejlesztők könnyen kialakíthatnak egy komplett fejlesztési környezetet a saját gépükön, amelyben tesztelhetik és finomíthatják a webalkalmazásokat.

Jelen esetünkben, mi a XAMPP-ot nem lokális gépen használtuk, hanem a phpMyAdmin-t összeköttöttük a szerverünkkel.

2.8. API

Az API az "Application Programming Interface" rövidítése. Az API egy olyan szoftver interfész, amely lehetővé teszi a különböző szoftveralkalmazások közötti kommunikációt és adatcsere lehetőségét.

Az API-k általában előre meghatározott szabványokat és protokollokat használnak, amelyek lehetővé teszik az alkalmazások közötti adatátvitelt. Az API-k lehetővé teszik, hogy az alkalmazások különféle funkciói és adatai elérhető legyenek más alkalmazások számára, hogy azok használni tudják őket. Például egy webhely, mint például a Facebook, rendelkezik egy API-val, amely lehetővé teszi a harmadik feleknek, hogy az alkalmazásukba integrálják a Facebook bejelentkezési rendszert vagy a Facebook felhasználók adatait.

Az API-k használata széles körben elterjedt a szoftverfejlesztésben, és lehetővé teszi az alkalmazások közötti integrációt, ami növeli az alkalmazások funkcionalitását és használhatóságát.

3. Adatbázis dokumentáció

3.1. Az adatbázis célja

Az adatbázisunk célja, hogy a weboldalon regisztráló felhasználók adatait, és a weboldalon történő funkciókat kezeljük és eltároljuk. Az oldalunkon lehetőség van a játékok, illetve a felhasználók értékelésére, hozzászólásra, illetve a hozzászólások értékelésére is, ezért tartottuk nagyon fontosnak egy relációs adatbázis létrehozását.

3.2. IGDB API

Ahhoz, hogy az oldalunkon több mint kétszázezer játékot megjelenítsünk, keresnünk kellett egy olyan megoldást, ahol nem mi manuálisan visszük fel a saját adatbázisunkba a játékokról az adatokat.

Egy olyan adatbázis forrást kellett találnunk, amit ingyenesen is fel tudunk használni, így találtuk meg az Internet Game Database (IGDB) nevű oldalt.

Az IGDB-hoz tartozik egy Developer API dokumentáció, ami részletesen leírja, mit kell tenni ahhoz, hogy elérjük az adatbázist. Egy Twitch fiókon keresztül kellett kérnünk egy Tokenet, amit az IGDB oldalán kellett regisztráltatni, hogy megkapjuk a hozzáférést. Az adatbázisukhoz való hozzáférést Postman-ben végeztük, ahol SQL-hez hasonló lekérdezéseket hajtottunk végre pl.:

```
fgame.name, url, game.genres.name, game.first_release_date, game.summary,  
game.platforms.slug, game.platforms.name, game.total_rating; sort total_rating_count  
desc; where game.platforms.slug = "win" & game.total_rating >= 90; 1 500;
```

Ezzel a lekérdezéssel minden 90-nél többre értékelt játékról lekérünk adatokat, mint a nevét, kategóriáját, megjelenési dátumát, leírását. Pontosan mi sem tudhatjuk, hogy hány játék található meg az oldalunkon, mivel a játékok száma napról napra változik, de nagyjából 230.000 játékkal dolgozunk.

A dokumentáció részletesen végigmegy a különböző lekérdezési módokon.

field	type	description
age_ratings	Array of Age Rating IDs	The PEGI rating
aggregated_rating	Double	Rating based on external critic scores
aggregated_rating_count	Integer	Number of external critic scores
alternative_names	Array of Alternative Name IDs	Alternative names for this game
artworks	Array of Artwork IDs	Artworks of this game
bundles	Array of Game IDs	The bundles this game is a part of
category	Category Enum	The category of this game
checksum	uuid	Hash of the object
collection	Reference ID for Collection	The series the game belongs to
cover	Reference ID for Cover	The cover of this game

18. ábra IGDB API mezők

Az API-nak korlátai is vannak, mivel az API-t szolgáltató cég nem szeretné, hogy túlságosan le legyen terhelve a saját szervere. A korlátok a következők: egyszerre 500 játék adatait kérhetjük le és másodpercenként 4 különböző lekérdezést folytathatunk egy oldalon.

API-ból lekért mezők, amiket használunk az oldalon:

id	Játékhoz tartozó egyedi azonosító
cover	Játék borítója
name	Játék neve
release_dates	Játék megjelenési ideje
game_modes	Játék módja
genres	Játék kategóriája
involved_companies	Játékfejlesztő cég és kiadója
platforms	Minden platform, amire a játék megjelent
websites	Játékhoz tartozó weboldalak
summary	Játékhoz tartozó leírás
videos	Játékhoz tartozó videók
screenshots	Játékhoz tartozó képek
rating	Játékhoz tartozó értékelés az IGDB oldala szerint
follows	Játékhoz tartozó játékosok általi követés az IGDB oldala szerint
dlcs	Játékhoz tartozó letölthető tartalmak

3.3. Szükséges információk

- Felhasználó adatai: felhasználónév, e-mail cím, jelszó, profilkép (avatar)
- Sessions: A regisztrált felhasználók alkalmankénti bejelentkezését kezeli.
- Felhasználó értékelése: A felhasználók értékelhetik egymás profilját az oldalon történő aktivitásuk alapján, illetve magukat a játékokat is értékelhetik.
- Hozzászólások értékelése: A felhasználók hozzászólhatnak a játékokhoz, illetve egymás profiljához. A játék, illetve a profil kommentelés külön táblában szerepel.

3.4. Kapcsolatok meghatározása

user => sessions: 1:N-es kapcsolat, mivel egy felhasználóhoz több session is tartozhat egy időben, de fordítva ez nem így van, egy sessionhoz csak egy felhasználó tartozhat.

user => comment: 1:N-es kapcsolat, mivel egy felhasználóhoz több komment tartozhat, egy hozzászólásnak csak egy írója lehet.

user => ratingTable: 1:N-es kapcsolat, mivel egy felhasználóhoz több értékelés tartozhat, de egy értékelés adott egyed-előfordulása csak egy felhasználóé lehet.

user => ratios: 1:N-es kapcsolat, mivel egy felhasználóhoz több értékelés (like, dislike) tartozhat, de egy értékeléshez csak egy felhasználó tartozhat.

komment => ratios : 1:N-es kapcsolat, mivel egy kommenthez szintén több értékelés tartozhat, de az értékelés egy adott egyed-előfordulása nem tartozhat több hozzászóláshoz.

lists <=> games: N:M kapcsolat, mivel egy listához több játék tartozhat és ez fordítva is igaz, egy játékhoz több lista is tartozhat. Felbontottuk két darab 1:N-es kapcsolatra.


lists => listGames: 1:N kapcsolat, a list egy egyed-előfordulásához a listGames több egyed-előfordulása tartozhat, de fordítva ez nem igaz.

games => listGames: 1:N kapcsolat, a games egy egyed-előfordulásához a listGames több egyed-előfordulása tartozhat, de fordítva ez nem igaz.

3.5.Users tábla

A felhasználói adatok tárolásáért felel.

mező	típus	leírás
ID	int	Egyedi azonosító, ami folyamatosan növekvő szám.
nev	varchar(255)	Felhasználónév, kötelező kitölteni a regisztráció során.
avatar	varchar(255)	A felhasználó által beállítható kép, amely a felhasználó profiljánál jelenik meg, és ide a kép linkjét lehet feltölteni.
email	varchar(50)	E-mail cím, kötelező kitölteni a regisztráció során.
jelszo	varchar(255)	Jelszó, amit a regisztráló megad, ez titkosítva kerül az adatbázisba biztonsági okok miatt. Kötelező kitölteni.
steam	varchar(200)	A felhasználó megadhatja a Steam profil linkét, amit itt tárolunk el.

id	nev	avatar	email	jelszo	steam
65	golyo	https://cdn.pixabay.com/photo/2015/	golyo@golyo.golyo	965d529ea8c8793c9f72bd507df084b6	
64	rujo	https://tr.rbxcdn.com/24015a966107	rujo@rujo.rujo	e410a050b926b3da0b4672e91dc090ce	
63	noobie.dookie	https://cdn.pixabay.com/photo/2015/	dookie@noobie.poopie	0a9174af1ad200e8a09616c731db2e88	
62		https://cdn.pixabay.com/photo/2015/	kobe@kobe.kobe	2357e8fb9945f0a2039a7093422a3dee	
61	kakoma	https://cdn.pixabay.com/photo/2015/	kakoma@kakoma.kakoma	301a323530477f0471b6bff4495d9782	
60	csiga	https://cdn.pixabay.com/photo/2015/	csiga@csiga.csiga	facca0cfef4633750a20382574b4422b	
59	muky	https://yt3.googleusercontent.com/i_	muky@muky.muky	df7c056ced24ea940ed93f8de5024272	
58	lupi	https://staging.cohostcdn.org/attachr	lupi@lupi.lupi	6dd0e57350be63c0486b8a9769696fcb	
57	busz	https://kep.cdn.index.hu/1/0/4727/47	busz@busz.busz	2372fa2d895ed4921d13a6110b0a7d3d	
56	bobby	https://cdn.pixabay.com/photo/2015/	bobby427766@gmail.com	a9c4cef5735770e657b7c25b9dcb807b	
49	árviztűró tukorfűrógép	https://cdn.pixabay.com/photo/2015/	masodik@masodik.masodik	19248fa849638e84819f18f7d39a1fe7	https://steamcommunity.com/id/GOBLIN

19. ábra Users tábla

3.6. Sessions tábla

Eltárolja a felhasználók munkameneteit.

mező	típus	leírás
ID	int	Egyedi azonosító, ami folyamatosan növekvő szám.
UserID	int	A felhasználó azonosítója, akihez az adott session tartozik.
active	tinyint(1)	Az active mező arra szolgál, hogy a felhasználó munkamenete éppen aktív-e, tehát nem járt-e le.
token	varchar(255)	Egy egyedi érték, ami azonosítja a munkamenetet.
acquired	datetime	A session kezdési időpontját jelenti, amikor például a felhasználó bejelentkezik a profiljába.
expires	datetime	A session lejáratási időpontját jelenti.

id	userID	active	token	acquired	expires
26	9	0	7ecee0322143e526957cbf838837197da431e48c	2023-03-14 13:15:10	2023-03-14 13:16:10
27	9	0	2f3fb1cf5c731c411b607387e0934ee10c651f5b	2023-03-14 13:18:55	2023-03-14 13:19:55
28	9	0	961b708492d5d9122ba9caa3071b162dee580143	2023-03-14 13:22:48	2023-03-14 13:32:48

20. ábra Sessions tábla

3.7.ratingTable

Ebben a táblában a felhasználók és a játékok értékelésére vonatkozó sorokat tároljuk.

mező	típus	leírás
ID	int	Egyedi azonosító, ami folyamatosan növekvő szám.
ratedThing	int	Az adott értékelt játék vagy felhasználó azonosítója, a type mezőtől függ, hogy ez melyikre vonatkozik

ratedBy	int	Ki értékelte az adott felhasználót vagy játékot.
type	varchar(10)	Típus, ami vagy user vagy game.
rating	int	Maga az értékelés jelenik meg ebben a mezőben, 1-10-ig terjed.


id	ratedThing	ratedBy	type 0 = user 1 = game	rating
127	119429	13	game	10
128	104671	13	game	5
129	83	13	user	8
130	82062	13	game	10
131	82	13	user	6
132	120933	13	game	8

21. ábra Rating tábla

3.8.Comment tábla

Ebben a táblában a hozzászólásokhoz kapcsolódó sorokat tároljuk.

mező	típus	leírás
ID	int	Egyedi azonosító, ami folyamatosan növekvő szám.
madeBy	int	Ki írta az adott a hozzászólást.
madeOn	int	Az adott értékelt játék vagy felhasználó azonosítója, a type mezőtől függ, hogy ez melyikre vonatkozik
type	varchar(10)	Típus, ami vagy user vagy game.
date	datetime	Mikor írták az adott hozzászólást.
text	varchar (6000)	A hozzászólás szövege, maximum 600 karakter.

id	madeBy	madeOn	type	date ▾ 1	text
205	9	56	user	2023-04-19 07:56:06	ain zwei drei fir
204	9	56	user	2023-04-19 07:55:50	bobby muskle is here
202	9	114879	game	2023-04-18 14:04:52	
201	9	9	user	2023-04-18 13:55:46	Oj Alija Alija Ijljanima babo I tebi će leđa okr...
197	54	54	user	2023-04-18 13:14:42	Köszönöm.
196	54	54	user	2023-04-18 13:14:38	Létszi ne ír ilyeneket a profilom alá.
194	55	241247	game	2023-04-18 13:14:20	love it, someone loves me even if its in a game an...

22. ábra Comment tábla

3.9.Ratios tábla

Itt található a hozzászólások értékelése.

mező	típus	leírás
ID	int	Egyedi azonosító, ami folyamatosan növekvő szám.
commentID	int	Annak a kommentnek az azonosítója, amihez ez az értékelés tartozik.
userID	int	A felhasználó azonosítója.
ratio	tinyint(1)	Lehet (+1) és (-1). Ha pozitív, akkor az értékelés tetszik a felhasználónak, ha pedig negatív, akkor nem tetszik. Akkor lehet az értéke 0, ha az adott felhasználó visszavonja az értékelését, mivel ilyenkor nem törli a sort az adatbázisból.
date	datetime	Mikor történt az adott értékelés.

id	commentID	userID	ratio	date
79	47	4	-1	2023-03-28 13:47:19
80	48	13	1	2023-03-30 12:02:28
104	13	9	1	2023-03-30 11:40:50
106	12	9	-1	2023-04-13 12:54:48
107	56	9	-1	2023-03-29 13:38:18

23. ábra Ratios tábla

3.10. lists tábla

mező	típus	leírás
ID	int	Egyedi azonosító, ami folyamatosan növekvő szám.
userID	int	A felhasználó azonosítója, akihez a lista tartozik.
nev	varchar(200)	Lista neve.
type	varchar(12)	Lista típusa.
visibility	tinyint(1)	Lista láthatósága, lehet privát vagy publikus (0 vagy 1).
order	int	Milyen sorrendben jelenik meg a lista.
updated	datetime	Mikor frissült utoljára a lista.

id	▼ 1	userID	nev	type	visibility	order	updated
233		62	playing	playing	1	2	2023-04-20 14:13:27
232		62	completed	completed	1	3	2023-04-20 14:13:27
231		62	wishlist	wishlist	1	4	2023-04-20 14:13:27
230		62	favorite	favorite	1	5	2023-04-20 14:13:27
229		59	muky vagyok	custom	1	0	2023-04-20 14:03:43
228		57	zseb	custom	0	0	2023-04-20 14:00:31

24. ábra Lists tábla

3.11. listGames tábla

Kapcsolótábla a games és a lists között

mező	típus	leírás
ID	int	Egyedi azonosító, ami folyamatosan növekvő szám.
listID	int	A lista egyedi azonosítója.
gameID	int	A játék azonosítója.
date	datetime	Mikor került be a listába az adott játék.

id	listID	gameID	date
76	176	18564	2023-04-19 10:31:58
80	157	14012	2023-04-19 10:47:12
81	158	14012	2023-04-19 10:47:47

25. ábra listGames tábla

3.12. games tábla

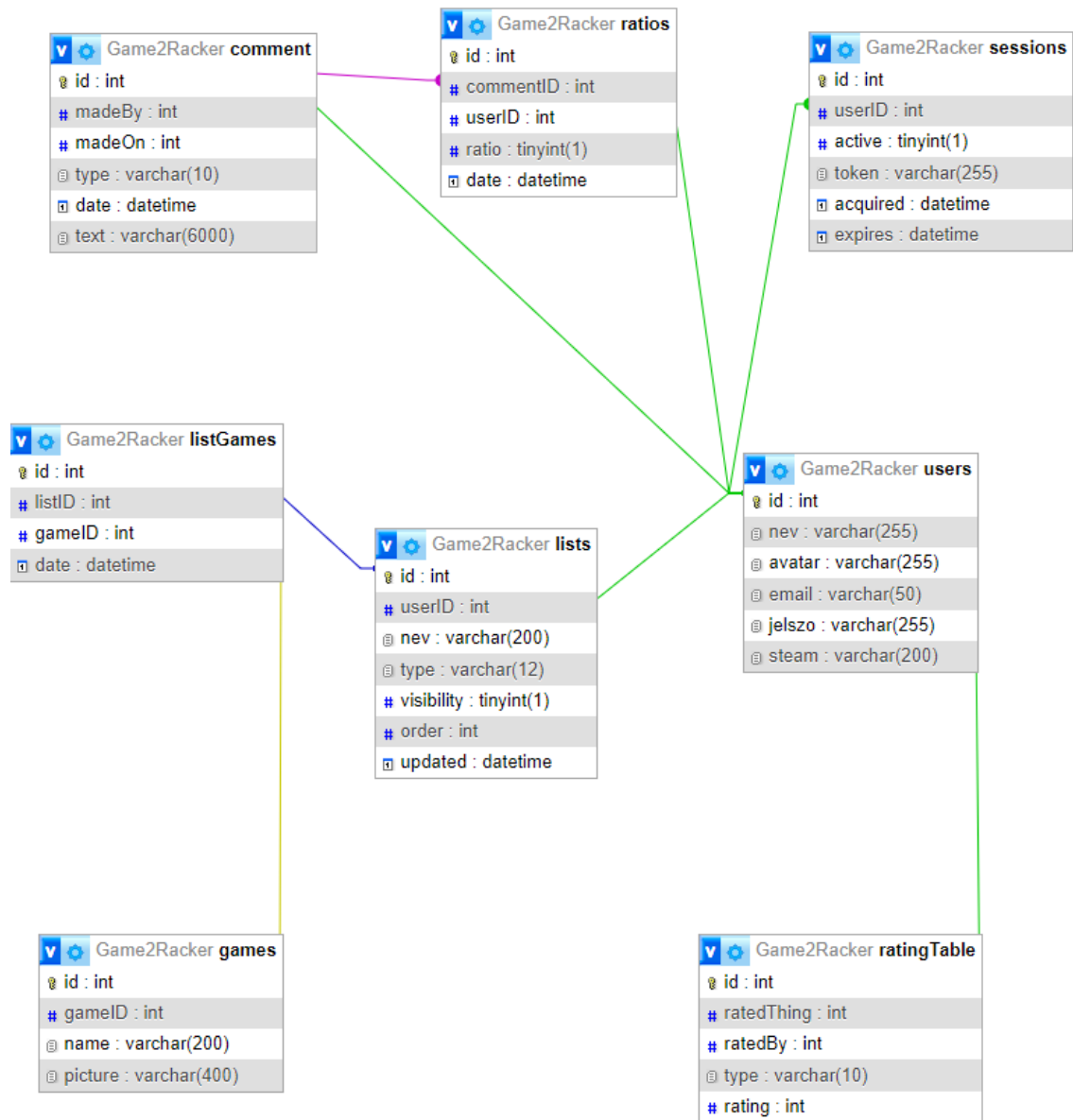
mező	típus	leírás
ID	int	Egyedi azonosító, ami folyamatosan növekvő szám.
gameID	int	A játék egyedi azonosítója. Ez a játék azonosítója az IGDB adatbázisban.
name	varchar(200)	A játék neve.
picture	varchar(400)	A játék képe.

id	gameID	name	picture
2	14012	Stardew Valley	https://images.igdb.com/igdb/image/upload/t_1080...
3	202171	Vampire Survivors	https://images.igdb.com/igdb/image/upload/t_cover_...
4	274396	Vampire Survivors Legacy of the Moonspell	https://images.igdb.com/igdb/image/upload/t_cover_...
5	297839	Vampire Survivors Tides of the Foscari	https://images.igdb.com/igdb/image/upload/t_cover_...

26. ábra games tábla

3.13. Adatbázis diagram

Egy 8 táblából álló adatbázist hoztunk létre.



27. ábra Adatbázis diagram

4. Fejlesztői dokumentáció

4.1. Globális fájlok

A projektünk kiindulópontja a var/www mappában található. Ezen belül a depend almappában találhatóak kiegészítő /template fájlok, melyek önmagukban nem képeznek működő oldalt, hanem más fájlok hivatkoznak rájuk.

Az oldalon az összes fájl függ a connection.php és a tokenHandler.php-től. Ezek tartalmát, funkcionalitását legtöbb alkalommal a navbar.php file segítségével kapjuk meg. Tehát a navbar.php elején ezen fájlok becsatolásra kerülnek.

```
require("connection.php");  
require("tokenHandler.php");
```

A connection.php fájlban definiálunk négy változót, amiket átadunk a mysqli_connect() php módszernek paraméterként, majd ennek visszatérési értékét eltároljuk egy ötödik változóban.

Ezt a változót fogjuk használni az összes adatbázisból való lekérdezésnél. Ha hiba történt az adatbázishoz való csatlakozáskor, akkor ez a változó FALSE értéket kap és megjelenít az oldalon egy hibaüzenetet.

```
$hostname = "87.229.6.116";  
$username = "bruhaju";  
$password = "Zobratyuff";  
$dbname = "Game2Racker";  
$conn = mysqli_connect($hostname, $username, $password, $dbname);  
if(!$conn){  
    echo "Database connection error".mysqli_connect_error();  
}
```

4.1.1. tokenHandler.php

Oldalunkra való sikeres bejelentkezéskor a „sessions” táblában egy új sort hozunk létre, ebbe a sorba kerülnek a bejelentkezett felhasználó munkamenetének tulajdonságai. Emellett létrehozunk sütit is, amely tartalmazza az előbb említett tulajdonságok közül magát a token-t (a munkamenet egyedi azonosítóját, ez biztonsági szempontból egy randomizált hexadecimális kód), tartalmazza még a munkamenet és egyben a süti lejárat dátumát.

```
$bytes = random_bytes(20);
```

```

$token = bin2hex($bytes);
$newId = $row['id'];
$stmt = $conn->prepare("INSERT INTO `sessions`(`userID`, `active`, `token`, `expires`)
VALUES (?,true,?,DATE_ADD(CURRENT_TIMESTAMP(), INTERVAL 3 DAY));");
$stmt->bind_param('ss', $row['id'],$token);
$stmt->execute();
$stmt->reset();
setcookie("session", $token, time() + (60*60*24*7), "/");

```

A TokenHandler.php első soraiban lefuttatunk egy lekérdezést, ami invalidálja az összes tokent, aminek a lejárat dátuma régebbi, mint a UNIX idő a lekérdezés pillanatában.

```

$queryAll = "UPDATE `sessions` SET `active`='0' WHERE expires < '".date("Y-m-d
H:i:s").".'";
mysqli_query($conn, $queryAll);

```

Lekérdezzük a felhasználó összes adatát, kivéve a jelszót, azzal a feltétellel, hogy ha a sessions táblában létezik olyan sor, amiben egyszerre az aktív mező igaz értéket tárol, a token mező egyed-előfordulása egyezik a felhasználó böngészőjében beállított süti tartalmával.

```

$tokenQuery = "SELECT
`users`.id,`users`.nev,`users`.avatar,`users`.email,`sessions`.id,`sessions`.userID,`sessions`.active,`sessions`.token,`sessions`.acquired,`sessions`.expires
FROM `users` RIGHT JOIN `sessions` on `sessions`.userID = `users`.id
WHERE `sessions`.token = ?
AND active = '1'";
$stmt = $conn->prepare($tokenQuery);
$stmt->bind_param('s', $_COOKIE["session"]);
$stmt->execute();
$result = $stmt->get_result();

```

E függvény segítségével egy „userData” nevű változónak értéket adunk. Ha az előbbi lekérdezés visszatér pontosan egy sorral, akkor a változó megkapja a lekérdezés eredményét tömb formában, ellenkező esetben egy üres tömböt ad át.

```

function getUserData($result){
    if ($result->num_rows === 1) {
        return $result->fetch_array();
    }
}

```

```

    }else {
        return array();}
}
$userData = getUserData($result);

```

4.1.2. navbar.php

Mivel a navbar fájlban meghívjuk az előbb bemutatott két fájlt, ezért a userData változóban eltárolt jelen belépett felhasználó adatai elérhetőek lesznek az összes olyan oldalon, amiben importálva van require() függvénnyel a navigációs sáv.

A navigációs sávunk tartalma a „userData” változó tartalmától függ, ha üres tömböt kapott értékül, a felhasználó a bejelentkezés és regisztráció gombokat fogja látni, ha sikeresen értéket kapott, akkor a felhasználó neve és avatarja jelenik meg.

Eltároljuk egy session változóban, hogy éppen hol vagyunk. Ez a bejelentkezésnél fontos, hogy miután végeztünk a bejelentkezéssel, oda kerüljünk, ahol rá kattintottunk, ne a főoldalra.

```

session_start();
$_SESSION['current_page'] = $_SERVER['REQUEST_URI'];

```

4.1.3. curl.php

A cURL a client URL rövidítése. A PHP cURL egy könyvtár, amely a PHP bővítménye. Lehetővé teszi a felhasználó számára a HTTP kérések létrehozását PHP-ben. A cURL könyvtárat arra használják, hogy más szerverekkel kommunikáljanak számos protokoll segítségével.

A cURL lehetővé teszi a felhasználó számára, hogy adatokat küldjön és fogadjon az URL szintaxis segítségével. A cURL segítségével egyszerűbbé válik a kommunikáció a különböző weboldalak és domainek között.

Az IGDB API-ja cURL technológiát használ az adatbázisukból való lekérdezésekre. Ezeket az adatokat json formátumban küldi vissza.

Egy curl függvényt használtunk az összes apihoz kapcsolódó lekérdezéshez.

```

function getData($url = "", $postFields = "")
{
    $curl = curl_init();
    curl_setopt_array($curl, array(

```

```

CURLOPT_URL => $url,
CURLOPT_RETURNTRANSFER => true,
CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10,
CURLOPT_TIMEOUT => 0,
CURLOPT_FOLLOWLOCATION => true,
CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
CURLOPT_CUSTOMREQUEST => 'POST',
CURLOPT_POSTFIELDS => $postFields,
CURLOPT_HTTPHEADER => array(
    'Client-ID: 1gdsndj0p60xua0ac6a6n7kcldeho8',
    'Authorization: Bearer im5fl0se5llzvdjv4py7eyvbsnaijk',
    'Content-Type: text/plain',
    'Cookie: __cf_bm=i6Ct3Bu80LLr8boRVUD1wxvgulosh8XFKTwuawwkfJQ-1678262079-0-AS34GswIhE0/1Ex1L6yfy/Av/Bcd19Ks+BwZ4W9MqR2TanALgtaOjKY62QITvCm+BpUKZUqUeQ+J7autpnE0eo4='
),
));
$response = curl_exec($curl);
curl_close($curl);
return $response;}

```

4.1.4. comments.php

Get metódussal lekérjük az url-ben utazó adatokat, ezek alapján értéket adunk a \$pageData tömbnek. Jelenleg az oldalunkon két komment típust különítünk el: játék vagy felhasználó oldalon lévő.

```

if (isset($_GET['gameid']) || isset($_GET['userid'])) {
    $pageData = array("type" => "", "id" => "");
    if (isset($_GET['gameid'])) {
        $pageData['type'] = "game";
        $pageData['id'] = $id;
    }
}

```

```

}
else if (isset($_GET['userid'])) {
    $pageData['type'] = "user";
    $pageData['id'] = $userid;
}

```

A pageData tömb tartalmazni fogja a felhasználó által aktuálisan használt oldal azonosítóját, melyet az alábbi lekérdezésekben használjuk.

A pageData alapján lekérdezzük az aktuális oldalra írt összes komment mellé egyenként adható értékelések (like, dislike) eredményét, amelyet a ratios nevű táblában tárolunk. Egy komment egy egyed-előfordulásához tartozó összes sort lekérjük a ratios.commentID nevű idegen kulcs alapján LEFT JOIN parancs segítségével. (Hogy azok a kommentek is megjelenjenek, amikhez nem tartozik értékelés). A tárolt kommenteket egyedi azonosítója (comments.id) alapján GROUP BY paranccsal csoportosítjuk, majd a SUM függvény használatával megjelenítjük a kommenthez tartozó komment-értékeket, ami az adott hozzászólás értékelését mutatja.

IFNULL függvénnyel oldottuk meg, hogy minden komment megjelenjen még akkor is, ha nem tartozik hozzá értékelés, ilyenkor „0” fog megjelenni.

```

$commentsSQL = "SELECT
    comment.id AS 'comment_id'
,comment.madeBy,comment.madeOn,comment.type,comment.date,comment.text,
`users`.id AS 'user_id' , `users`.nev, `users`.avatar, IFNULL(SUM(ratios.ratio),0) as
ratio

FROM `comment` LEFT JOIN `users` ON comment.madeBy = `users`.id
LEFT JOIN `ratios` ON ratios.commentID = comment.id

WHERE comment.type = '{ $pageData['type'] }' AND comment.madeOn =
'{ $pageData['id'] }' GROUP BY comment.id ORDER BY comment.date DESC;";

$queryComments = mysqli_query($conn, $commentsSQL);

```

4.1.5. rating.php

A működéséhez szükségünk van a navbar.php-ra.

Lekéri az adott játék vagy user értékeléseit, és átlagolja. Ezek után értékül adja egy változónak, ha nincs még értékelés rajta, akkor 0-t ad vissza.

```
$avgRatingSQL = "SELECT ROUND(AVG(rating),2) as avgrating FROM `ratingTable`  
WHERE ratedThing = '{$pageData['id']}' AND type = '{$pageData['type']}';";  
  
$avgRatingRes = mysqli_query($conn, $avgRatingSQL);  
  
$avgRatingRow = mysqli_fetch_array($avgRatingRes);  
  
if ($avgRatingRow['avgrating'] != NULL) {  
    $avgRating = $avgRatingRow['avgrating'];  
} else {  
    $avgRating = "0";  
}
```

Lekérdezi a saját értékelést, és ha van, akkor a yourRating változónak adja az értéket, ha pedig nincs, akkor ugyanennek a változónak egy üres stringet ad értékül.

```
$yourRatingSQL = "SELECT * FROM `ratingTable` WHERE ratedThing =  
{ $pageData['id'] } AND ratedBy = { $userData['userID'] } AND type =  
'{ $pageData['type'] }'";  
  
$yourRatingRes = mysqli_query($conn, $yourRatingSQL);  
  
if (mysqli_num_rows($yourRatingRes) > 0) {  
    $yourRatingRow = mysqli_fetch_array($yourRatingRes);  
    $yourRating = $yourRatingRow['rating'];  
} else {  
    $yourRating = "";  
}
```

4.1.6. lists.php

A működéséhez szükségünk van a navbar.php-ra.

A „getGameData” függvénynek paraméterként megadjuk az aktuális játék azonosítóját.

Ez alapján létrehozunk egy tömböt, ami tartalmazni fogja a játék azonosítóját, nevét és a hozzá tartozó képet. Ezután meghívjuk JavaScript Fetch API használatával a „getUsersLists.php” fájlt, ami lekéri a felhasználó összes listáját, felépíti a listaelemek html szerkezetét, és visszaküldi egy php változóban, ezt be is illeszti az oldalra.

Ezután hozzátcsolunk egy eseményfigyelőt kattintásra, hogy az elemek a felhasználó által interaktálhatóak legyenek.

```
function getGameData(id) {  
    listMenu.classList.remove("hidden");  
    gameData = [];  
    let name = "";  
    let picture = "";  
    <?php if (isset($id)) { ?>  
        name = <?php echo "\"$gameName\"" ?>;  
        picture = <?php echo "\"$gamePic\"" ?>;  
    <?php }else { ?>  
        name = document.getElementById(id).childNodes[2].firstElementChild.innerText;  
        picture = document.getElementById(id).childNodes[1].firstElementChild.src;  
    <?php }?>  
    gameData.push(id,name,picture);  
    let dataForPHP = new FormData();  
    dataForPHP.append("id", gameData[0]);  
    fetch(`depend/getUsersLists.php`, {  
        method: "POST",  
        body: dataForPHP  
    })  
    .then(response => response.text())  
    .then(data => {  
        listMenuContainer.innerHTML = data;  
        listItems = document.querySelectorAll(".list-menu-item-clicker");  
        for (let i = 0; i < listItems.length; i++) {  
            listItems[i].addEventListener('click',() =>  
addToList(listItems[i],gameData[0],listItems[i].id,gameData[1],gameData[2]));  
        }  
        refreshKukaList();  
    })  
}
```



```
.catch(error => console.log(error));
}
```

Listához való hozzáadásnál JavaScript Fetch API használatával meghívjuk a pushLists.php fájlt, átadjuk a „gameData” tömböt.

Lekérdezi, hogy az átadott játék benne van-e az adatbázisban, ha nincs, akkor feltölti.

Két lekérdezéssel folytatja először, hogy a lista tulajdonosa egyezik-e felhasználóval, aki a listához adást végrehajtotta. Másodszor, hogy a játék szerepel-e már az adott listában.

Ha már szerepel, akkor kiveszi a listából, különben hozzáadja.

```
$isListTheUsersSQL = "SELECT * FROM `lists` WHERE userID =
{$userData['userID']} AND id = {$listID}";

$isGameOnListSQL = "SELECT * FROM `listGames` WHERE gameID = {$gameID}
and listID = {$listID}";

$isGameInDbSQL = "SELECT * FROM `games` WHERE `gameID` = ?";

$stmt = $conn->prepare($isGameInDbSQL);

$stmt->bind_param('s',$gameID);

$stmt->execute();

$result = $stmt->get_result();

$stmt->reset();

if($result->num_rows < 1){

    $putGameInDbSQL = "INSERT INTO `games`(`gameID`, `name`, `picture`)
VALUES (?, ?, ?)";

    $stmt = $conn->prepare($putGameInDbSQL);

    $stmt->bind_param('sss',$gameID,$gameName,$gamePic);

    $stmt->execute();

    $stmt->reset();

}

if (mysqli_num_rows(mysqli_query($conn, $isListTheUsersSQL)) > 0 &&
mysqli_num_rows(mysqli_query($conn, $isGameOnListSQL)) == 0) {

    $pushListGamesSQL = "INSERT INTO `listGames`(`listID`, `gameID`) VALUES
(?, ?)";

    $stmt = $conn->prepare($pushListGamesSQL);
```

```

$stmt->bind_param('ss',$listID,$gameID);

$stmt->execute();

$stmt->reset();

echo "INSERTED";
}

else if(mysql_num_rows(mysql_query($conn, $isListTheUsersSQL)) > 0 &&
mysql_num_rows(mysql_query($conn, $isGameOnListSQL)) == 1){

    $deleteGameFromListSQL = "DELETE FROM `listGames` WHERE `listID` = ?
AND `gameID` = ?";

    $stmt = $conn->prepare($deleteGameFromListSQL);

    $stmt->bind_param('ss',$listID,$gameID);

    $stmt->execute();

    $stmt->reset();

    echo "REMOVED";
}

```

4.2. Regisztráció

A regisztráció megvalósításához szintén a PHP-t választottuk, mert így könnyedén létre tudunk hozni egy egyszerűen működő regisztrációs űrlapot. Az űrlapban a regisztrálni kívánó személyeknek meg kell adniuk a felhasználónevüket, e-mail címüket, a jelszót, illetve a beírt jelszót meg is kell nekik erősíteni, hogy biztosan jól írták-e be, ezzel elkerüljük azt, hogy esetleg egy rossz jelszó megadása esetén a felhasználónak úgy kell találgatnia a megadott jelszó után.

A kódunk először ellenőrzi, hogy az űrlapot elküldték-e a "submit" gomb megnyomásával. Ha igen, akkor a kód feldolgozza a felhasználó által megadott adatokat, és ellenőrzi, hogy a megadott e-mail cím vagy felhasználónév már létezik-e az adatbázisban. Ha létezik, akkor hibaüzenetet jelenít meg, és nem engedi a felhasználónak a regisztrációt. Ha nincs ilyen felhasználó, akkor ellenőrzi a megadott jelszavak egyezőségét. Ha a jelszavak egyeznek, akkor hozzáadja az új felhasználót az adatbázishoz.

A kód használja a POST módszert a bejelentkezési adatok elküldéséhez a szervernek, biztonságosabbá teszi az adatokat az SQL Injection támadások elleni védelem érdekében. A jelszó md5 hash-el, ami biztonságosabb, mint az egyszerű, szöveges jelszó tárolása, bár a hash függvények ma már nem javasoltak jelszavak hash-elésére, de jelenlegi ismereteink alapján választottuk ezt a titkosítási módszert.

Az űrlap tartalmazza a regisztrációhoz szükséges mezőket, és egy "Register now" gombot a regisztráció elküldéséhez. Ha a felhasználó már rendelkezik fiókkal, akkor az "Already have an account?" felirat alatt található linkre kattintva átirányítják őt a bejelentkezési oldalra.

```
<?php
require("../depend/connection.php");
if(isset($_POST['submit'])){
    $name = htmlspecialchars($_POST['name']);
    $email = htmlspecialchars($_POST['email']);
    $pass = md5($_POST['password']);
    $cpass = md5($_POST['cpassword']);
    $select = "SELECT * FROM `users` WHERE email = ? OR nev = ?";
    $stmt = $conn->prepare($select);
    $stmt->bind_param('ss', $email,$name);
```

```

$stmt->execute();
$result = $stmt->get_result();
$stmt->reset();
if($result->num_rows > 0){

    $error[] = 'User already exist!';

}else{

    if($pass != $cpass){
        $error[] = 'Password not matched!';
    }else{
        $insert = "INSERT INTO `users`(nev, email, jelszo) VALUES(?,?,?)";
        $stmt = $conn->prepare($insert);
        $stmt->bind_param('sss', $name,$email,$pass);
        $stmt->execute();
        $stmt->reset();
        echo $name;
        header("Location: login.php");
        exit();
    }
}

};
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Game2Racker</title>
    <link rel="stylesheet" href="../css/login.css">

```

```

</head>
<body>
<div class="form-container">
  <form action="" method="post">
    <h3>Register now</h3>
    <?php
    if(isset($error)){
      foreach($error as $error){
        echo '<span class="error-msg">'.$error.'</span>';
      };
    };
    ?>
    <input type="text" name="name" required placeholder="Enter your name">
    <input type="email" name="email" required placeholder="Enter your email">
    <input type="password" name="password" required placeholder="Enter your
password">
    <input type="password" name="cpassword" required placeholder="Confirm your
password">
    <input type="submit" name="submit" value="Register now" class="form-btn">
    <p id="account">Already have an account? <a href="login.php">Login
now</a></p>
  </form>
</div>
</body>
</html>

```

4.3. Bejelentkezés

Létrehoztuk a már említett bejelentkezési felületet is, hogy az oldalon történő hasznos funkciókat a felhasználók könnyedén tudják használni. A bejelentkezésnél a kódunk használ egy MySQL adatbázist, amit egy kapcsolatfájl (connection.php) segítségével ér el.

A megírt script a bejelentkeztetési adatokat (e-mail és jelszó) ellenőrzi, majd ha azok helyesek, akkor létrehoz egy munkamenetet (session), amelyet a sessions táblában tárolunk. Az aktív munkamenet azonosítására egy token-t generál, amelyet a felhasználó

böngészőjébe helyezett süti (cookie) tárol. Az aktív munkameneteknek egy lejáratási idejük van, amelyet 3 napra állítottunk. A munkamenet azonosítója és a felhasználó ID-je szintén a sessions táblában van tárolva.

```
if(isset($_POST['submit'])){

    $name = mysqli_real_escape_string($conn, $_POST['name']);
    $email = mysqli_real_escape_string($conn, $_POST['email']);
    $pass = md5($_POST['password']);

    $select = "SELECT * FROM `users` WHERE email = ? && jelszo = ?";
    $stmt = $conn->prepare($select);
    $stmt->bind_param('ss', $email,$pass);
    $stmt->execute();
    $result = $stmt->get_result();
    $row = $result->fetch_assoc();
    $stmt->reset();

    if($result->num_rows == 1){
        $bytes = random_bytes(20);
        $token = bin2hex($bytes);
        $newId = $row['id'];
        $stmt = $conn->prepare("INSERT INTO `sessions`(`userID`, `active`, `token`, `expires`) VALUES (?,true,?,DATE_ADD(CURRENT_TIMESTAMP(), INTERVAL 3 DAY));");

        $stmt->bind_param('ss', $row['id'],$token);
        $stmt->execute();
        $stmt->reset();

        setcookie("session", $token, time() + (60*60*24*7), "/");
    }
}
```

Ha a sikeres bejelentkezés megtörténik, akkor a script átirányítja a felhasználót az aktuális oldalra (\$_SESSION['current_page']), amelyet a munkamenet létrehozása előtt azonosítottak.

4.4. Játék oldal

A játékok lekérdezéséhez el kell érni az IGDB API-át, amit a curl.php-ban írtunk meg. Az url nevű változó itt a covers táblában keres az API-n belül és a „dataQuery” változóban adjuk meg a lekérdezést.

```
$url = "https://api.igdb.com/v4/covers";  
$dataQuery =  
'fgame.id,game.name,url,game.first_release_date,game.summary,game.platforms.slug,g  
ame.platforms.name,game.genres.name,game.involved_companies.company.name,gam  
e.videos.video_id,game.game_modes.name, game.screenshots.url,game.websites.url;  
where id = ' . $id . ' ; 1 1;'  
$CurledData = getData($url, $dataQuery);
```

A „game_data” változónak értékül adjuk az asszociatív tömbbé alakított „curledData” json értékű változót.

```
$game_data = json_decode($CurledData);
```

Egy „foreach” ciklusban végigmegyünk a „game_data” változón és a tömb elemeire hivatkozunk. Egy „echo” segítségével megjelenítünk minden számunkra szükséges adatot a játékhoz.

```
foreach ($game_data as $game) {  
    $gameName = $game->game->name;  
    echo "<div class = \"flex\">  
  
    <div class = \"flex2\">  
  
    <div class = \"mediaflex\">  
        <h1>\" . $gameName . \"</h1>\";  
        $timestamp = $game->game->first_release_date;
```

```

if ($timestamp == "") {
    echo "";
}
else {
    echo "<p style = \"font-weight: bold;\">Release Date</p>";
    echo gmdate("Y.m.d", $timestamp);
}
echo "<p style = \"font-weight: bold;\">Game mode(s)</p>";
foreach ($game->game->game_modes as $modes) {
    echo "<p>" . $modes->name . "</p>";
}
echo "<p style = \"font-weight: bold;\">Genres</p>";
foreach ($game->game->genres as $genre) {
    echo "<p>" . $genre->name . "</p>";
}
echo "<p style = \"font-weight: bold;\">Developer(s) &
Publisher(s)</p>";
foreach ($game->game->involved_companies as $company) {
    echo "<p>" . $company->company->name . "</p>";
}

```

Hasonló módon valósítottuk meg a lekéréseket, adatok megjelenítését a search.php és index.php fájlokban.

4.5. Felmerült problémák a fejlesztés során

Munkáink során nagyon sok akadályba ütköztünk. Először is egy teljesen más projektet álmodtunk meg a csapattal, amit el is kezdtük megvalósítani. Ez volt a Gather2Watch, ami egy úgymond együtt-néző alkalmazás létrehozása volt. Elég jól haladtunk vele az elején, aztán amikor következett a Nodejs használata sajnos rájöttünk, hogy ismeretek hiányában más projektet kell választanunk.

Amikor a felhasználó kattint a like és a dislike gombokra, akkor eltelik egy kis idő az interakció és annak az adatbázisba kerülése között. Ha rövid időn belül megismétli a felhasználó a kattintást, akkor mivel az adatbázisba még nem került bele az adat, a szelekció figyelmen kívül hagyja az eseményt, és az előző összes végrehajtott interakció is felkerül időközben. Végül ezt a problémát sikerült orvosolni.

Összefoglalás

A projektmunkánk előtt kitűzött céljainkat a legnagyobb százalékban sikerült megvalósítanunk. Azok a funkciók, amelyeket még tervezünk továbbfejleszteni, a következők:

Barátlista, barátok hozzáadásának lehetősége, elfogadása és elutasítása, ahhoz hasonlóan, ahogyan a többi közösségi média oldalakon is működik.

- Steam API használata a steam profillal való hatékonyabb összekötése érdekében.
- Alaplisták (pl: kedvencek, kívánságlista) létrehozása, létező felhasználókhoz való hozzárendelése. A listához hozzáadása gomb mellett megjeleníteni őket külön gombként.
- Külön oldal, amiben listák tartalma jelenne meg részletesebben.
- Listák személyre szabható sorrendjének profilon való megjelenése.
- Lehetőség listák átnevezésére.
- Lehetőség felhasználónév cserére.
- E-mail rendszer telepítése és beaktiválása a szerverünkön.
- Regisztrációkor fiók aktiválásához szükséges megerősítő e-mail.
- Új jelszó kérésének lehetősége e-mailben elfelejtett jelszó esetén.

Amikor már az oldalunk olyan mennyiségű látogatót tud felmutatni, amely alapján a Google engedélyezi a reklámjainak az elhelyezését, ezekből bevételre is szert tudunk

tenni, amiből majd finanszírozhatjuk a szerverbérletet, illetve a további fejlesztői munkáinkat is. Ebből fakadóan született meg egy másik továbbfejlesztési ötletünk is, a VIP regisztráció, ami annyit jelentene, ha valaki hajlandó egy jelképes összeget fizetni, akkor nem jelennének meg a profiljába belépve a felületen a reklámok.

Továbbá a VIP regisztrációval rendelkezőknek lehetőségük lenne különböző avatarokat választani, melyekre a normál regisztrációval rendelkezőknek nincs lehetőségük. Egy VIP jelvény is beépítésre kerülne, így a felhasználók láthatnák, hogy például az adott felhasználó VIP regisztrációval rendelkezik, mikor megnézik egymás profilját. Így viszont majd egy fizetési rendszert is ki kell alakítani, hogy pl. PayPal-on keresztül tudjanak fizetni a felhasználók a reklámmentes felületért, illetve a VIP regisztrációért.

Mivel mindannyian folytatjuk tanulmányainkat ezen a területen, minden bizonnyal a későbbiekben fogunk hozzá készíteni mobil applikációt is.

Projektünk elérhetősége: <https://github.com/DrSzklenar/Game2Racker>

Weboldal elérhetősége: <http://game2racker.felx.hu>

Források

1. IGDB Internet Game Database: <https://api-docs.igdb.com/#getting-started>
2. Stack Overflow: <https://stackoverflow.com/questions/10995294/border-radius-not-working>
3. Stack Overflow: <https://stackoverflow.com/questions/4366730/how-do-i-check-if-a-string-contains-a-specific-word>
4. Stack Overflow: <https://stackoverflow.com/questions/44961886/how-to-get-parent-div-data-attribute-javascript>
5. Stack Overflow: <https://stackoverflow.com/questions/70339213/get-dataset-of-parent-while-clicking-on-child>
6. Stack Overflow: <https://stackoverflow.com/questions/60174/how-can-i-prevent-sql-injection-in-php>
7. Vanilla Kinetic: <https://github.com/dzek69/vanilla.kinetic>
8. Stack Overflow: <https://stackoverflow.com/questions/6014702/how-do-i-detect-shiftenter-and-generate-a-new-line-in-textarea>
9. Stack Exchange: <https://dba.stackexchange.com/questions/44956/good-explanation-of-cascade-on-delete-update-behavior>

Ábrajegyzék

1. ábra Navigációs sáv	5
2. ábra Főoldalon megjelenő játékok	6
3. ábra Videó és képek a játékról	7
4. ábra Egy játék adatai	7
5. ábra Listák	8
6. ábra Keresés	8
7. ábra Keresési eredmények	9
8. ábra Profil menü	9
9. ábra Profil kép és score	10
10. ábra Felhasználó profilja	10
11. ábra Felhasználó listái	10
12. ábra Beállítások	11
13. ábra Regisztráció	11
14. ábra Bejelentkezés	12
15. ábra ÁSZF	13
16. ábra Lábléc (footer)	13
17. ábra Hozzászólások	14
18. ábra IGDB API mezők	20
19. ábra Users tábla	22
20. ábra Sessions tábla	23
21. ábra Rating tábla	24
22. ábra Comment tábla	25
23. ábra Ratios tábla	25
24. ábra Lists tábla	26
25. ábra listGames tábla	27
26. ábra games tábla	27
27. ábra Adatbázis diagram	28