



ZÁRÓDOLGOZAT

Készítették:

Jancsurák Bence

Borbély Zoltán

Holácsik Judit

Konzulens:

Farkas Zoltán

Miskolc

2023.

Miskolci SZC Kandó Kálmán Informatikai Technikum

Miskolci Szakképzési Centrum

Az 5-0613-12-03 számú Szoftverfejlesztő- és tesztelő szak

ZÁRÓDOLGOZAT

Project Galaxy

Jancsurák Bence - Borbényi Zoltán - Holácsik Judit

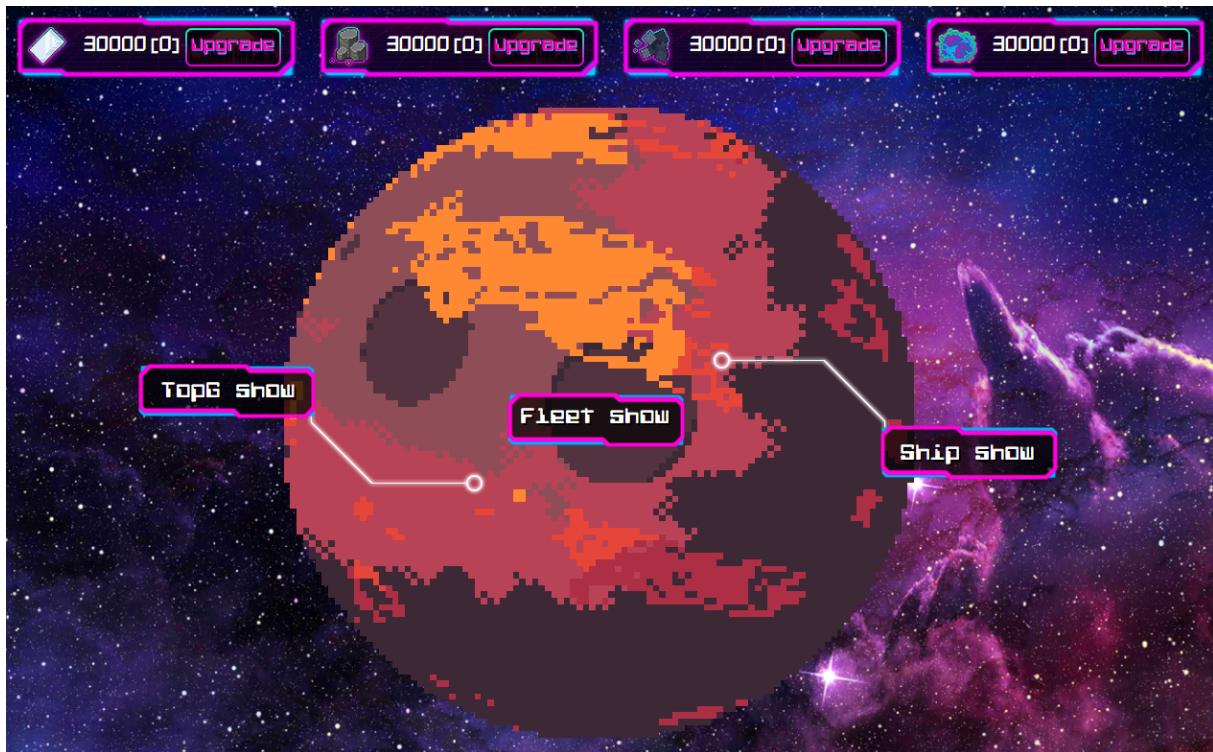
2022 - 2023

TARTALOM

A PROJEKTRÓL.....	5
FRONTEND.....	6
HTML:.....	6
CSS:.....	6
React:.....	7
Bootstrap:.....	7
Axios:.....	8
Js-Cookie:.....	8
Moment-js:.....	8
BACKEND.....	9
Asp.NET Core 6.0 Web API:.....	9
C#:.....	9
Entity Framework Core:.....	10
MailKit:.....	10
XAMPP:.....	10
MYSQL:.....	11
Lighthouse:.....	11
JWT TOKEN:.....	12
ADATBÁZIS SZERKEZET.....	13
FELHASZNÁLT PROGRAMOK.....	15
Visual Studio 2022:.....	15
Visual Studio Code:.....	15
CSAPATMUNKA.....	16
Github:.....	16
Messenger:.....	16
Discord:.....	17
TESZTELÉS.....	18
Frontend tesztelés Lighthouse segítségével:.....	18
Backend tesztelés Unit tesztekkel:.....	18
AZ ALKALMAZÁS.....	19
Bejelentkezés:.....	19
Regisztráció:.....	20
Bolygó választás:.....	20
UI:.....	21
Fleet show:.....	18
Ship creation:.....	18
Planet info:.....	21
Battle:.....	22
Exploration:.....	23
Messages:.....	24
FORRÁS.....	25

A PROJEKTRŐL

A csoportunk a projektünk témájának egy bögészős bolygó menedzsment játéket választottunk, amelyben a játékosnak a feladata, hogy egy erőteljes galaktikus birodalmat építsen fel. A játék során a játékosnak lehetősége van arra, hogy saját bolygóját fejlessze és növelje a gazdagságát, miközben hajókat épít és felfedezőutakra küld, hogy feltérképezze a környező területeket és forrásokat gyűjtsön. A játékosoknak lehetőségük van arra is, hogy támadásokat indítsanak más bolygók ellen amiben összemérhetik erejüket más játékosokkal. A játék izgalmas kihívásokat és lehetőségeket kínál az űrkalandok rajongóinak, és a játékosoknak lehetősége van arra, hogy bizonyítsák a stratégiai és taktikai képességeiket a galaxis uralása érdekében.



1. ábra - A bolygónk és a köri lötte lévő UI elemek

FELHASZNÁLT TECHNOLÓGIÁK ÉS PROGRAMOK

FRONTEND

HTML:

A HTML az angol HyperText Markup Language rövidítése, ami magyarul hiperszöveges jelölőnyelvet jelent. Weboldalak készítéséhez kifejlesztett nyelv, ami az oldal szövege mellett különböző leíró elemeket tartalmaz, amik blokkokba rendezik a tartalmat, illetve olyan plusz elemekkel egészítik ki, mint például képek, videók vagy űrlapok. Mindent tag-ek közé kell írni pl.: <h1> ami a nyitó tag az egyes szintű fejléc szövegért, majd </h1> záró tag-gel lezárjuk.

Amikor megnyitsz egy weboldalt, a böngésző a webszerverről letölt egy HTML formátumban megírt szöveges fájlt (ez a weboldal tartalma), illetve általában egy CSS fájlt is, ami azt határozza meg, hogy ez a tartalom pontosan hogyan nézzen ki. A böngésző futtatja ezt az állományt, majd kirakja a képernyőre.

CSS:

Az angol Cascading Style Sheets rövidítése, melynek szószerinti fordítása: "Egymásba ágyazott stíluslapok". A CSS egy olyan számítástechnikai nyelv, amivel a weboldalak kinézetét lehet megadni, például hogy milyen színű legyen egy honlap háttere és mekkora betűk jelenjenek meg rajta.

A CSS első verziója 1996-ban jelent meg. Megalkotásának elsődleges célja az volt, hogy a weboldalak tartalmát (azaz azt, hogy mi jelenjen meg) **elkülönítsék a külalakjától** (azaz attól, hogy hogyan jelenjen meg). Az olvasott szövegek két helyről előhívva jelenik meg: a HTML tartalmazza magukat a mondatokat, és a CSS adja meg, hogy ezek a mondatok pl: fehér alapon fekete színű betűkkel jelenjenek meg, sőt azt is, hogy ez aláhúzott, *ez pedig dőlt* legyen. Ennek köszönhetően egyszerűbb megváltoztatni a weboldalak arculatát. Ha például kék színben szeretnénk megjeleníteni ezt a szöveget, csupán csak egy helyen, egyetlen sort kellene átírni a kódban ahelyett, hogy mindenhol kiadnánk a parancsot, hogy a betűk kékek legyenek.

React:

React egy nyílt forráskódú JavaScript könyvtár, amelyet felhasználói felületek (UI) készítésére használnak. A Facebook fejlesztette ki, és ma már egy nagy közösség által karbantartott technológia.

React segítségével újra felhasználható komponenseket lehet létrehozni, amelyek önállóan működnek, és összeállíthatók, hogy összetett felhasználói felületeket hozzanak létre. React deklaratív megközelítést alkalmaz az UI készítése során, ami azt jelenti, hogy a fejlesztők leírják, hogy az UI hogyan kell kinéznie és hogyan kell viselkednie, ahelyett, hogy manuálisan manipulálnák a DOM-ot.

React egyik kulcsfontosságú jellemzője a virtuális DOM használata. A virtuális DOM egy könnyűsúlyú reprezentációja a tényleges DOM-nak, amely lehetővé teszi, hogy React hatékonyabban frissítse az UI-t. Amikor egy komponens állapota megváltozik, a React frissíti a virtuális DOM-ot, majd összehasonlítja a korábbi virtuális DOM-mal. Ezután alkalmazza a szükséges változásokat a tényleges DOM-ra, ami eredményez egy hatékonyabb és reaktívabb felhasználói felületet.

React-t széles körben használják a webfejlesztésben, és ez egy népszerű választás az összetett, dinamikus felhasználói felületek készítéséhez. Gyakran használják más könyvtákkal és keretrendserekkel, mint például a Redux az alkalmazás állapotának kezeléséhez vagy a Next.js a szerveroldali rendereléshez.

Bootstrap:

A Bootstrap lehetővé teszi a fejlesztők számára, hogy gyorsan és hatékonyan tervezzenek és fejlesszenek weboldalakat és webalkalmazásokat. A Bootstrap előre elkészített HTML, CSS és JavaScript komponenseket kínál, amelyek könnyen testre szabhatók és integrálhatók a weboldalba vagy alkalmazásba. Ezek a komponensek olyan alapvető elemeket tartalmaznak, mint a gombok, az űrlapok, a navigációs menük, a modális párbeszédpanelek és sok más. A Bootstrap keretrendszer további előnye, hogy nagy hangsúlyt helyez a reszponzív designra, azaz a weboldalak és alkalmazások automatikus alkalmazkodására a különböző kijelzőméretekhez és készülékekhez. A Bootstrap kompatibilis a legtöbb modern JavaScript keretrendszerrel és könyvtárral, mint például a React, a Vue.js és az Angular, így könnyen integrálható a meglévő projektekre. A Bootstrap-t széles körben használják a webfejlesztésben.

Axios:

Az Axios egy nyílt forráskódú JavaScript könyvtár, amelyet a böngésző- és szerveroldali HTTP kérések kezelésére használnak. Az Axios használható a modern böngészőkben és a Node.js szerveroldali környezetben is.

Az Axios lehetővé teszi a fejlesztők számára, hogy egyszerű és hatékony módon kommunikáljanak az API-val. A könyvtár nagyon intuitív interfészét kínál, amely lehetővé teszi a HTTP kérések egyszerű és gyors összeállítását, például GET, POST, PUT és DELETE kérések küldése. Az Axios támogatja a Promise API-t, így könnyen kezelhetők az aszinkron kérések és válaszok.

Az Axios egyéb funkciói közé tartozik a fejlesztőbarát hibakezelés, a kérések széles körű konfigurálhatósága, például a timeout kezelése, és a kérésekhez kapcsolódó adatok küldése, például az átadott adatok formátuma vagy a kérés header-jei.

Az Axios népszerű választás a modern JavaScript alkalmazások fejlesztése során, mivel egy könnyűsúlyú és megbízható megoldást kínál a HTTP kérések kezelésére. Az Axios kompatibilis számos modern JavaScript keretrendszerrel és könyvtárral, például a React, a Vue.js és az Angular.

Js-Cookie:

A cookie-k kezeléséhez használt npm package. Segít lementeni, beállítani a cookie-kat, sokat egyszerűsít a kódolásnál és gyorsít a kód lefolyásában.

Moment-js:

Nagyszerű npm package az idő kezelésére. Többféle beállítási módja van a kiírásra és időzítésre. Sokkal egyszerűbb, mint az alap idő kezelési lehetőségek. Beállítható bármilyen időzóna és a kiírásnál is különböző országok által használt módszereket lehet beállítani, illetve könnyen állítható, hogy mennyi adatot adjon meg. Pl csak év / hó / nap vagy akár ezred másodperces időkiírással is lehet. Akár 2 időpont közt eltelt időt is kiszámítja automatikusan.

BACKEND

Asp.NET Core 6.0 Web API:

Az ASP.NET Core 6.0 Web API egy keretrendszer a .NET fejlesztők számára, amely lehetővé teszi a hatékony és korszerű webes alkalmazások fejlesztését. A Web API-k olyan webalkalmazások, amelyek kommunikálnak más rendszerekkel, például mobilalkalmazásokkal vagy webalkalmazásokkal, adatok lekérdezésére, frissítésére vagy adatok előállítására.

Az ASP.NET Core 6.0 Web API lehetővé teszi a fejlesztők számára, hogy egyszerűen és hatékonyan fejlesszenek HTTP API-kat. A keretrendszer tartalmazza az alapvető komponenseket, amelyek szükségesek a Web API-k fejlesztéséhez, például az HTTP kezelést, a route-olást és a modell validációt.

Az ASP.NET Core 6.0 Web API további előnyei közé tartozik a magas teljesítmény, a skálázhatóság és a biztonság. A keretrendszer lehetővé teszi a fejlesztők számára, hogy különféle hitelesítési és azonosítási módszereket alkalmazzanak, például a JWT-t.

Az ASP.NET Core 6.0 Web API támogatja az OpenAPI specifikációt, amely lehetővé teszi a dokumentáció automatikus generálását és a kliensek számára könnyű API használatát. A keretrendszer széles körben használható a modern webalkalmazások fejlesztése során, és lehetővé teszi a fejlesztők számára, hogy gyorsan és hatékonyan készítsenek skálázható és biztonságos API-kat.

C#:

A C# a .NET keretrendszer egyik leghasználatabb objektumorientált programozási nyelve. Korábban a .NET alkalmazások csak Windows alatt voltak elérhetőek, azonban 2016-ban kiadásra került a .NET Core.

A .NET Core megjelenésével a Microsoft kiterjesztette a .NET alkalmazhatóságát MacOS és Linux rendszerekre is.

Entity Framework Core:

Az Entity Framework Core egy nyílt forráskódú, cross-platform ORM (Object-Relational Mapping) keretrendszer a .NET fejlesztők számára. Az Entity Framework Core lehetővé teszi a fejlesztők számára, hogy könnyen kezeljék az adatokat és az adatbázisokat a .NET alkalmazásaikban.

Az ORM egy olyan technológia, amely lehetővé teszi a fejlesztők számára, hogy a relációs adatbázisokat objektumorientált kódolással kezeljék. Az Entity Framework Core automatikusan generálja az adatbázis-sémát az alkalmazásban használt osztályok alapján, és lehetővé teszi az adatok lekérdezését, frissítését és törlését az adatbázisból.

Az Entity Framework Core lehetővé teszi a fejlesztők számára, hogy a .NET Core-ra, valamint az Entity Framework 6-ra épülő alkalmazásokat is készítsenek. Az Entity Framework Core támogatja a különféle adatbázis-motorokat, például az SQL Server, az SQLite, a PostgreSQL és az MySQL.

Az Entity Framework Core további előnyei közé tartozik a könnyű telepítés és használat, az egységes adatelérési interfész, a LINQ támogatás, a hatékony adatmódosítások kezelése, a szűk keresési kifejezések lehetősége, az általános hibakezelés és a tesztelhetőség. Az Entity Framework Core egy népszerű választás a .NET fejlesztők között, akik gyorsan és hatékonyan szeretnék kezelní az adatokat az alkalmazásaikban.

MailKit:

Levelezést segítő keretrendszer, amelyben megadhatunk egy SMTP szervert, amit használva a felhasználóknak küld levelet.

XAMPP:

Mysql adatbázis és az ahhoz tartozó apache szerver elindításában nyújt segítséget. Egyszerű használni és telepíteni. Localhost-on indítja el a szervert alapbeállításon port nélkül. Megkönnyíti a programozáshoz szükséges adatcserét.

MySQL:

MySQL egy nyílt forráskódú relációs adatbázis-kezelő rendszer (RDBMS), amely lehetővé teszi az adatok hatékony kezelését és tárolását. Az adatok táblákban vannak tárolva, és az adatbázis-kezelő rendszer segítségével lehet azokat lekérdezni, frissíteni, törlési vagy beszúrni.

A MySQL az egyik legnépszerűbb relációs adatbázis-kezelő rendszer a világon, és széles körben használják a különféle alkalmazásokban, például webalkalmazásokban, adatbázisokban, szoftverekben és még sok másban.

A MySQL nagyon skálázható és megbízható adatbázis-kezelő rendszer. A rendszer támogatja a transzakciókat, a referenciális integritást és az ACID (atómikus, konzisztens, izolált, tartós) tulajdonságokat, amelyek biztosítják az adatok integritását és konzisztenciáját.

Lighthouse:

Egy ingyenesen is használható chrome bővítmény, ami teszteli és ellenőrzi az oldal minőségét, sebességét. A végén pontozza az oldalt és felajánl az esetleges teljesítmény növekedéshez szükséges változtatási lehetőségeket, tippeket. A lighthouse tesztről többet majd a tesztelésnél olvashat.

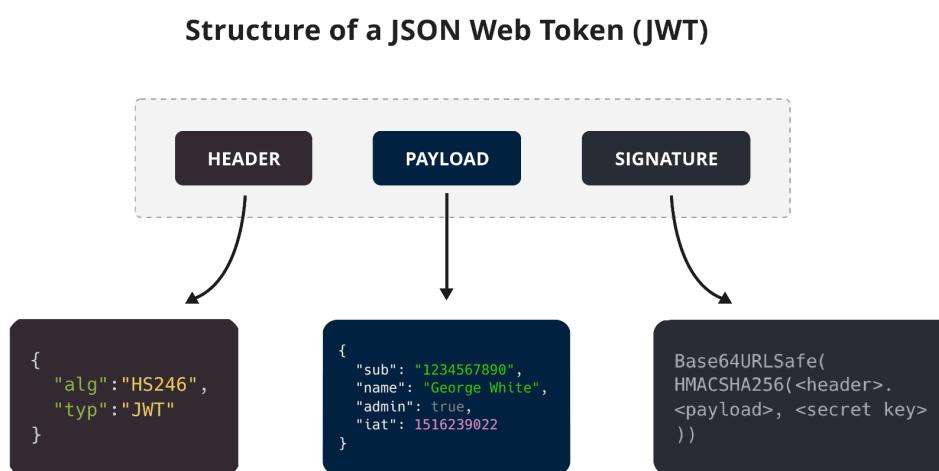
JWT TOKEN:

A JWT (JSON Web Token) egy nyílt szabvány, amely egy kódolt JSON objektumot használ a digitális azonosítás és hitelesítés céljára. A JWT tokeneket gyakran használják azonosításra és hitelesítésre a webes alkalmazásokban és API-kban, különösen az állapotmentes környezetben.

A JWT tokenek három részből állnak: az első rész a fejléc, a második a payload (tartalom), és a harmadik pedig a digitális aláírás. A fejléc a JWT típusát és az alkalmazott titkosítási algoritmust tartalmazza, a payload tartalmazza az adatokat, amelyeket a token hordoz, például a felhasználóazonosítót vagy a szerepkört, és a digitális aláírás pedig az adatok biztonságát és épségét garantálja.

A JWT tokeneknek számos előnye van a hagyományos autentikációs módszerekkel szemben. Az egyik legfontosabb előnyük, hogy a tokenek állapotmentesek, vagyis nincs szükség állapottárolásra a szerver oldalon, és így nagyban csökkenhető a szerveroldali terhelés. Emellett a tokenek nagyfokú biztonságot nyújtanak, mivel a digitális aláírásukkal ellenőrizhető a tartalmuk épsége és eredetisége.

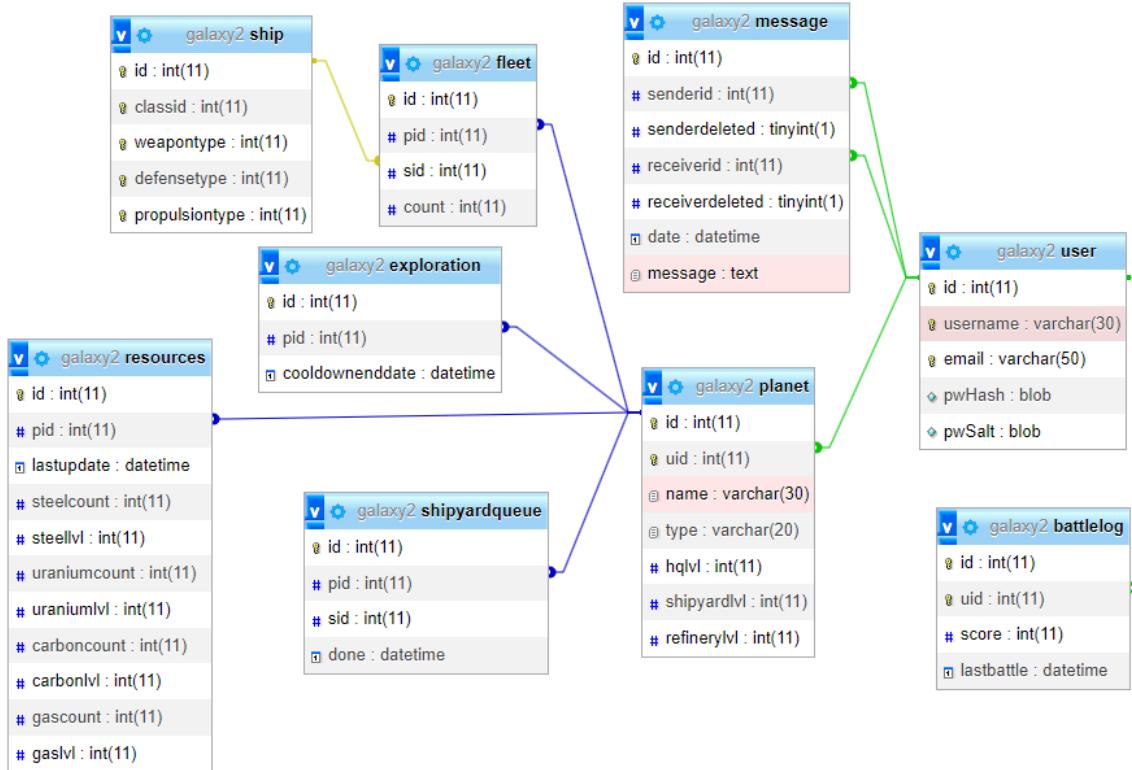
A JWT tokenek széles körben használatosak a modern webes alkalmazásokban és API-kban, és számos programozási nyelvben elérhetők, például a JavaScript-ben, a Python-ban vagy a Java-ban.



SuperTokens

2.ábra - JWT token működése

ADATBÁZIS SZERKEZET



3.ábra - Az adatbázis szerkezete

A ‘**user**’ tábla a felhasználói információkat tárolja, ideértve a felhasználónevet, az e-mail címet és a hashelt jelszót. A só használata a jelszó hashelése során növeli a biztonságot, mivel nehezebbé teszi a támadók számára a jelszó feltörését előre kiszámolt táblázatok vagy rainbow táblák segítségével. Az ‘**id**’ elsődleges kulcs lehetővé teszi más táblák hozzácsatolását.

A ‘**planet**’ tábla a bolygók adatait tárolja, beleérte a bolygó nevét, típusát és az épületek szintjét (főhadiszállás, hajógyártó és finomító). Ennek a táblának az ‘**id**’ elsődleges kulcsa jelenik meg idegen kulcsként a legtöbb táblában. Az ‘**uid**’ idegen kulccsal kapcsolódik a felhasználókhoz.

A ‘**resources**’ tábla az erőforrások adatait tárolja, ideértve az acél, urán, szén és gáz mennyiségét és azok bányászatához szükséges épületek szintjét. Az utolsó frissítés időpontját is tárolja, amely alapján az erőforrások mennyisége frissül. A ‘**pid**’ segítségével kapcsolódik egy adott bolygóhoz.

A ‘**fleet**’ tábla a flotta adatait tárolja, beleértve a flotta azonosítóját, a bolygó és a hajó azonosítóját és a flottában szereplő hajók számát. A ‘pid’ és ‘sid’ oszlopok külső kulcsok, amelyek kapcsolatot teremtenek a planet és ship táblákkal.

A ‘**ship**’ tábla tárolja a hajók adatait, beleértve az azonosítót és az osztály, fegyverzet, védelmi rendszer és hajtómű típusát. A classid, weapontype, defensetype és propulsiontype oszlopok kulcsok, amelyek a különböző modulokat kapcsolják az adott hajóhoz egy beégetett JSON fájl alapján, valamint ez a négy oszlop együtt egyedi, tehát egy kombináció csak egyszer szerepelhet a táblában.

A ‘**shipyardqueue**’ tábla tartalmazza azokat az adatokat, amelyek a hajógyárban folyamatban lévő hajók megépítéséhez szükségesek. Az id mező egyedi azonosítót tartalmaz, a pid mező a bolygó id-jét tárolja, amelyen a hajógyár található, a sid mező egy hajó id-jére utal, amely épül, a done mező pedig azt az időpontot rögzíti, amikor az adott hajó elkészül a gyárban.

Az ‘**exploration**’ tábla a felfedezésekhez tartozó adatokat tárolja. Az id mező egyedi azonosítót tartalmaz, a pid mező a játékos bolygójának id-jére mutat, a cooldownenddate mező pedig azt az időpontot rögzíti, amikor a játékos újabb felfedezést kezdhet el indítani.

Az ‘**battlelog**’ tábla a játékos harcokkal kapcsolatos adatait tárolja. Az id mező egyedi azonosítót tartalmaz, az uid mező a felhasználó azonosítójára mutat, amely egy idegen kulcs a user táblából. A score mező tartalmazza a játékos jelenlegi pontszámát, a lastbattle mező pedig az utolsó harc időpontját rögzíti.

Az ‘**message**’ tábla az üzeneteket tárolja. Az id mező egyedi azonosítót tartalmaz. A senderid mező az üzenet küldőjének azonosítójára mutat, a receiverid mező pedig az üzenet címzettjének azonosítójára mutat. Mindkét mező idegen kulcsok a user táblából. A senderdeleted és receiverdeleted mezők bit mezők, amelyek azt jelzik, hogy a felhasználó törölte-e az üzenetet. A date mező az üzenet dátumát és időpontját tartalmazza, a message mező pedig az üzenet szöveges tartalmát.

FELHASZNÁLT PROGRAMOK

Visual Studio 2022:

A Visual Studio 2022 egy integrált fejlesztői környezet (IDE), amelyet a Microsoft fejlesztett ki. Az IDE a szoftvertervezési, programozási, tesztelési és hibakeresési folyamatok támogatására szolgál. A Visual Studio 2022 számos programozási nyelvet támogat, beleértve a C#, C++, F#, Python, TypeScript és másokat.

Az új Visual Studio 2022 több új és fejlesztett funkcióval rendelkezik, amelyek célja, hogy segítsék a fejlesztőket hatékonyabban dolgozni, és elősegítsék a fejlesztési folyamat felgyorsítását. A Visual Studio 2022 új tervezőfelületet, frissített debuggert, fejlett kódkezelést, optimalizált kódolási élményt, frissített verziókezelést és még sok más funkciót tartalmaz.

A Visual Studio 2022 több platformon fut, beleértve a Windows, a macOS és az Ubuntu rendszereket is, így lehetővé teszi a fejlesztők számára, hogy bármilyen környezetben dolgozzanak. A Visual Studio 2022-t különféle fejlesztők és fejlesztőcsoportok használják világszerte, például az üzleti alkalmazások, a játékok, a mobilalkalmazások és a webes alkalmazások fejlesztéséhez.

Visual Studio Code:

A Visual Studio Code egy ingyenes, nyílt forráskódú kód szerkesztő, melyet a Microsoft fejleszt Windows, Linux és macOS operációs rendszerekhez. Támogatja a hibakeresőket, valamint beépített Git támogatással rendelkezik, továbbá képes az intelligens kódkiegészítésre az IntelliSense segítségével. A VSCode-ban a felhasználók megváltoztathatják a kinézetet (témat), megváltoztathatják a szerkesztő gyorsbillentyű-kiosztását, az alapértelmezett beállításokat és még sok egyebet. Támogatja a kiegészítőket, melyek segítségével további funkciók, testreszabási lehetőségek érhetők el.

CSAPATMUNKA

Github:

A GitHub egy olyan weboldal és felhőalapú szolgáltatás, amely segít a fejlesztőknek kódjuk tárolásában és kezelésében, valamint a kódjukban bekövetkezett változások nyomon követésében és ellenőrzésében. Egy profitorientált vállalat, amely felhőalapú Git-tárhely tárhely szolgáltatást kínál. Lényegében megkönnyíti az egyének és a csapatok számára a Git használatát verziókezelési és együttműködési célokra. A GitHub felülete elég felhasználóbarát ahhoz, hogy még a kezdő programozók is kihasználhassák a Git előnyeit. A GitHub nélkül a Git használata általában egy kicsit több technikai tudást és a parancssor használatát igényli. A GitHub azonban annyira felhasználóbarát, hogy egyesek még más típusú projektek például könyvek írása kezelésére is a GitHubot használják. Ráadásul bárki ingyenesen regisztrálhat és üzemeltethet nyilvános kódtárat, ami a GitHubot különösen népszerűvé teszi a nyílt forráskódú projektek körében.

Messenger:

A Messenger egy ingyenes üzenetküldési alkalmazás, melyet az amerikai Facebook fejlesztett ki. Tökéletes üzenetküldésre és fogadásra, ami nagyban megkönnyíti a kapcsolattartást kettő vagy akár több személy között is egyszerre. Lehetőségünk van hangüzenet küldésére, fogadására, valamint hívás, videohívásban való kommunikációra is. Küldhetünk képeket, videókat, szöveges dokumentumokat, de akár még különböző linkekkel is megoszthatunk egymással. Különböző hangulatjelekkel is kifejezhetjük véleményünket, érzéseinket egymás felé. Megalkotója Mark Zuckerberg, aki a Facebook-ot 2004. Februárjában indította útjára, míg a Messenger legelőször 2008-ban kezdte meg működését. Azóta az alkalmazás nem csak Android és Ios eszközökre érhető el, hanem számítógépekre is letölthető, használható.

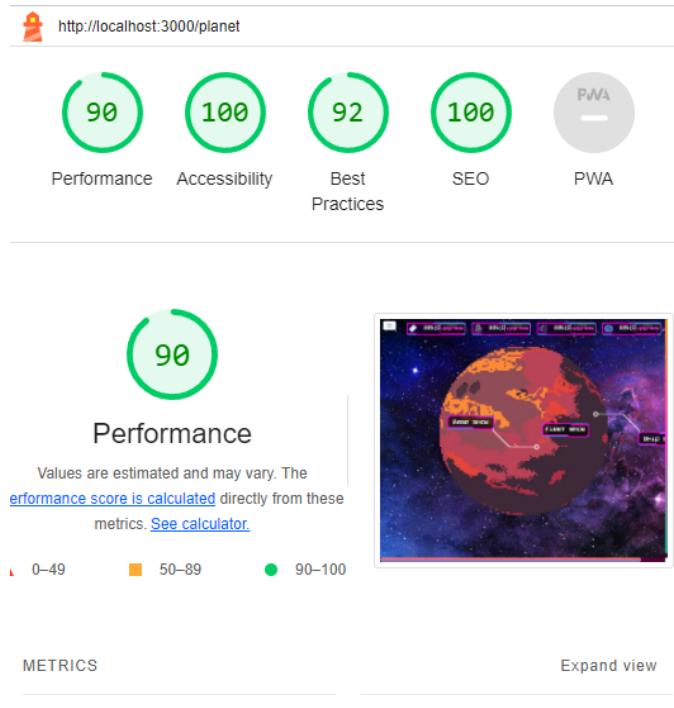
Discord:

A Discord egy első körben ingyenes internetes alkalmazás és digitális terjesztési platform, amit leginkább videójátékot játszó közösségek számára fejlesztettek ki, de bárki más is használhatja. Úgy terveztek, hogy nagy rendszerigényű programok, első körben játékok futtatásánál is problémák nélkül működjön. A platform rendelkezik szöveges, kép és videó, valamint audio kommunikációval is. Windows, MacOS, Android,iOS, Linux operációs rendszereken és böngészőkön fut. Alapötlete Jason Citrontól származik. Magán és az állami közösségek létrehozására és kezelésére készült. Ez a felhasználók számára hozzáférést biztosít a kommunikációs szolgáltatások köré összpontosító eszközökhöz, például a hang és videohívásokhoz, az állandó csevegőszobákhoz, a többi játékos központú szolgáltatással való integrációhoz, a közvetlen üzenetek küldésének és a személyes csoporthoz létrehozásának általános képességehez. Lehetőségünk van Nitro vásárlására is, amelyek segítenek áthidalni például az egyszerre való üzenetküldési limiteket, valamint segít nagyobb felbontású képek megosztásában is.

TESZTELÉS

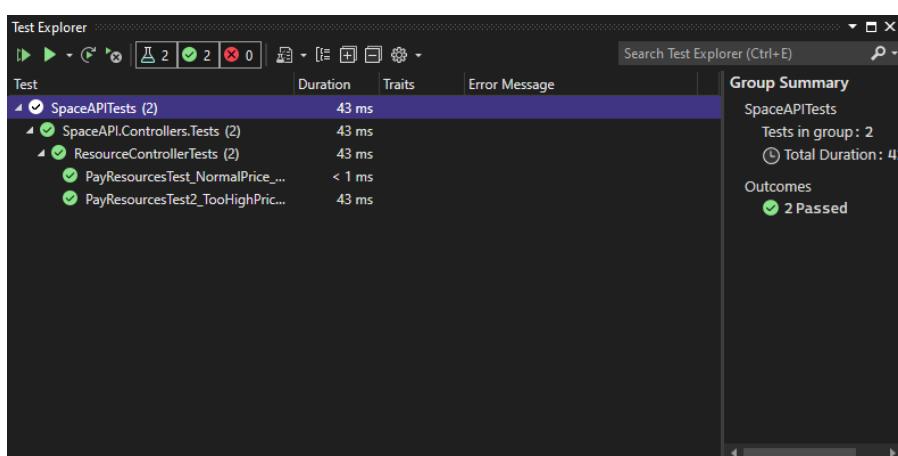
Frontend tesztelés Lighthouse segítségével:

Mint fent is említettük a lighthouse leírása során ez a bővítmény átvizsgálja az oldalt és ellenőrzi, hogy milyen a teljesítménye, hozzáférhetőség stb. Az alábbiakat dobta a fő oldalunkra:



4. ábra - Lighthouse tesztelés

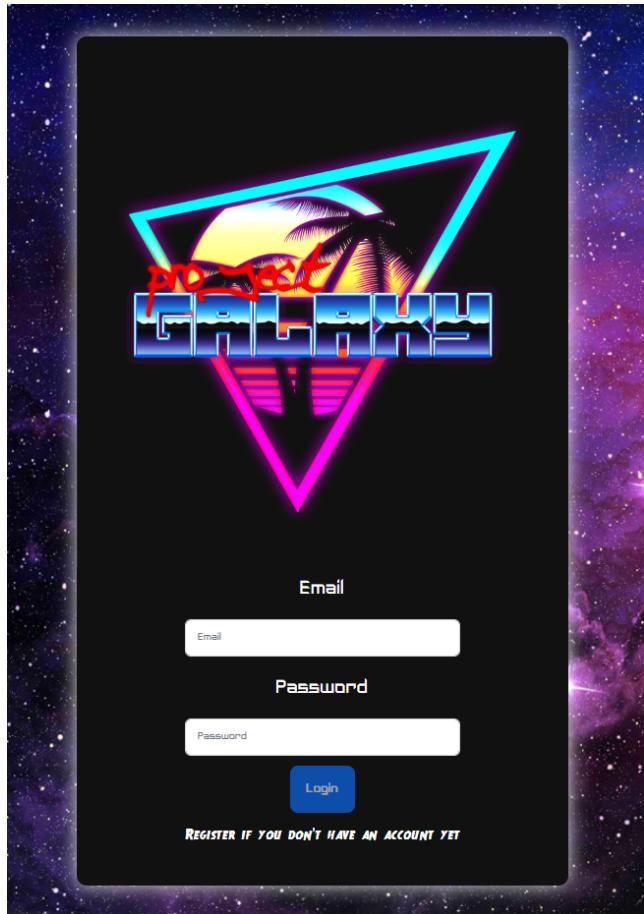
Backend tesztelés Unit tesztekkel:



AZ ALKALMAZÁS

Bejelentkezés:

A LoginPage egy olyan felületet biztosít, ahol a felhasználók bejelentkezhetnek a rendszerbe. A felület tartalmazza az e-mail és jelszó mezőket, amelyeket a felhasználónak



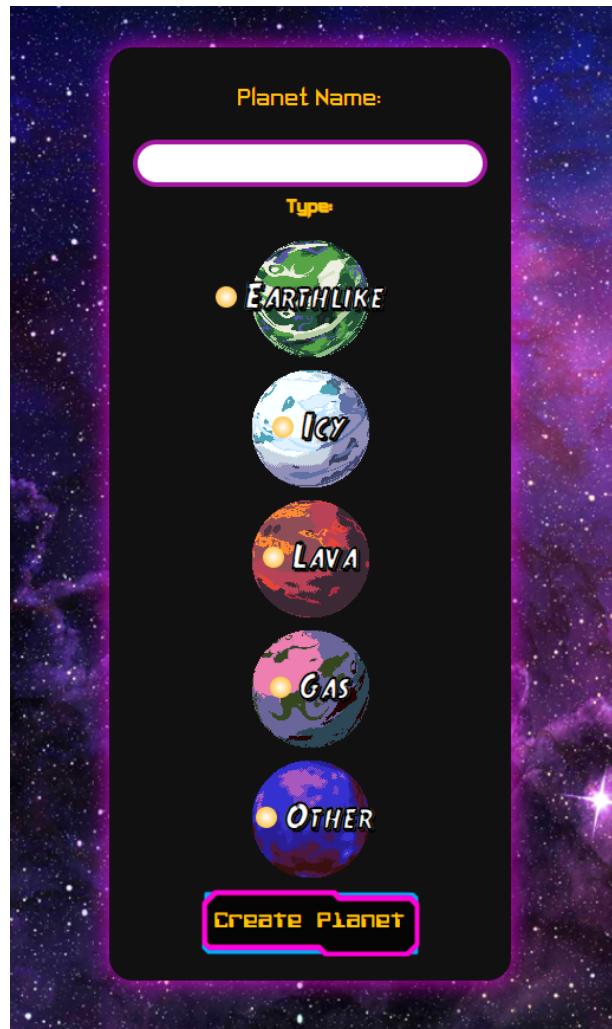
kitöltve lehet belépni a rendszerbe. Ha a felhasználó már bejelentkezett korábban és a token-je még érvényes, akkor a Logout gomb megjelenik a felületen, és a felhasználó kijelentkezhet az alkalmazásból. A bejelentkezési adatok helyes megadása esetén a felhasználó belép az alkalmazásba, és tovább navigál a PlanetPage oldalra. Ha a bejelentkezési adatok hibásak, akkor a felhasználó egy hibaüzenetet kap a képernyőn. A LoginPage komponenshez hozzá van rendelve egy Form komponens is, amely lehetővé teszi a felhasználók számára, hogy kitöltsék az e-mail és jelszó mezőket. A Form-on belül található egy link, amely lehetővé teszi a felhasználók számára, hogy regisztráljanak az alkalmazásba, ha még nem rendelkeznek fiókkal. A LoginPage komponens nagyon egyszerűen használható, és könnyen kezelhető.

Regisztráció:

A RegisterPage segítségével a felhasználó kitölthet egy formot a nevével, e-mail címével és jelszavával, majd regisztrálhat az alkalmazásba. Az űrlapban szereplő mezők kitöltése kötelező, különben a "Register" gomb nem lesz aktív és nem lehet regisztrálni. Az e-mail címnek érvényesnek kell lennie, különben az űrlap nem fogja elfogadni. A "Register" gombra kattintva az űrlap elküldi az adatokat a szervernek, ahol a felhasználó regisztrálódik. Ha bármilyen hiba történik, a felhasználó értesítést kap az űrlap tetején, és az űrlapot újra elküldheti a megfelelő adatokkal. Ha a regisztráció sikeres, a felhasználó átirányításra kerül a főoldalra.

Bolygó választás:

A PlanetCreatePage egy űrlap, ahol lehetőség van a felhasználónak egy saját bolygó létrehozására. Az oldalon található egy "Planet Name" mező, ahol meg kell adni a bolygó nevét, valamint egy "Type" rész, ahol a felhasználó kiválaszthatja a bolygó típusát a megadott lehetőségek közül. A típusok a következők: "Earthlike", "Icy", "Lava", "Gas" és "Other". A felhasználó az általa választott típust rádió gomb segítségével tudja kiválasztani. Az oldal alján található egy "Create Planet" gomb, amelyre kattintva az oldal elküldi a bolygó nevét és típusát az adatbázisba a megfelelő API hívás segítségével. Ha a létrehozás sikeres volt, a felhasználó bekerül a PlanetPage oldalra. Ha valamilyen hiba történik a bolygó létrehozása során, akkor az oldal hibajelzést jelenít meg a felhasználó számára.



UI:

Az oldal UI-ját avagy felhasználói felületét próbáltuk az űr témához megfelelően dizájnolni. A retrrowave világ irányába mentünk, mivel ez tökéletesen illett a sci-fi kinézethez. minden menüpontot próbáltunk egy dizájn sémára megcsinálni. Lehetőleg próbáltuk egyszerre reponszívá tenni böngészőben és telefonon is. A gombok megfelelő méretűek minden kettő részen és az olvashatóságra, felhasználható baráti környezetre is ügyeltünk. A legtöbb menü adja magát kezelés szempontjából, viszont a fő planet menüpontnál van egy kis eltérés. A háttérben a felhasználó által választott bolygó raktuk, ami fölé lévő div-ben helyeztünk három gombot, amelyek modal ablakokat nyitnak meg. Az oldal legtetején találhatóak a resource-ok és az annak a fejlesztési lehetőségei (upgrade gomb).

Planet:

Ez a játék központi oldala, amely tartalmazza a felhasználó adatait, a bolygóját, az erőforrásokat, a flottát és a hajógyártást. A komponensben található useEffect hookok segítségével az oldal betöltésekor lekérdezés történik a backend szerverről a felhasználó bolygójának, erőforrásainak, flottájának és hajóinak adatairól. Amennyiben az adatlekérdezés során a program nem talál bolygót, a felhasználó átirányításra kerül a bolygó létrehozása oldalra. A kódban található handleUpgradeResources és handleUpgradeBuildings függvények segítségével az erőforrásokat és a bolygó épületeit lehet fejleszteni. Amennyiben nem áll rendelkezésre elég erőforrás, a felhasználó figyelmeztetést kap. A handleAddShip függvény segítségével a felhasználó új hajókat adhatnak a flottához. Az adatok beillesztése után a backend ellenőrzi, hogy megfelelő mennyiségű erőforrás áll-e rendelkezésre a hajók építéséhez. Ha nem, a felhasználó figyelmeztetést kap. A komponens továbbá MainModal, FleetModal és ShipModal komponensekre mutató gombokat tartalmaz, amelyekkel segítségével a felhasználó megtekintheti a bázisát, flottáját és hajókat tud gyártani.

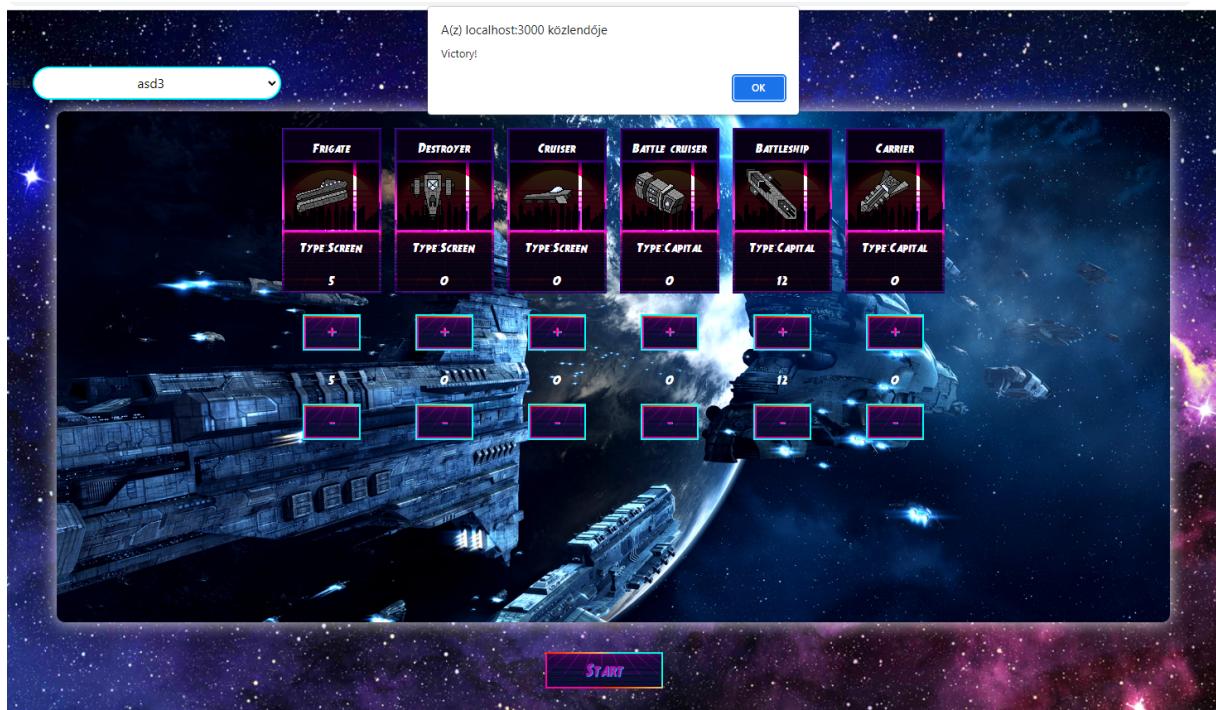
MainModal: Itt a bolygó és azon található építményeket lehet kezelni (szintjeit).

FleetModal: Itt lehet megnézni a jelenlegi hajókat és hogy milyen fegyver, shield és propulsion engine van rajta.

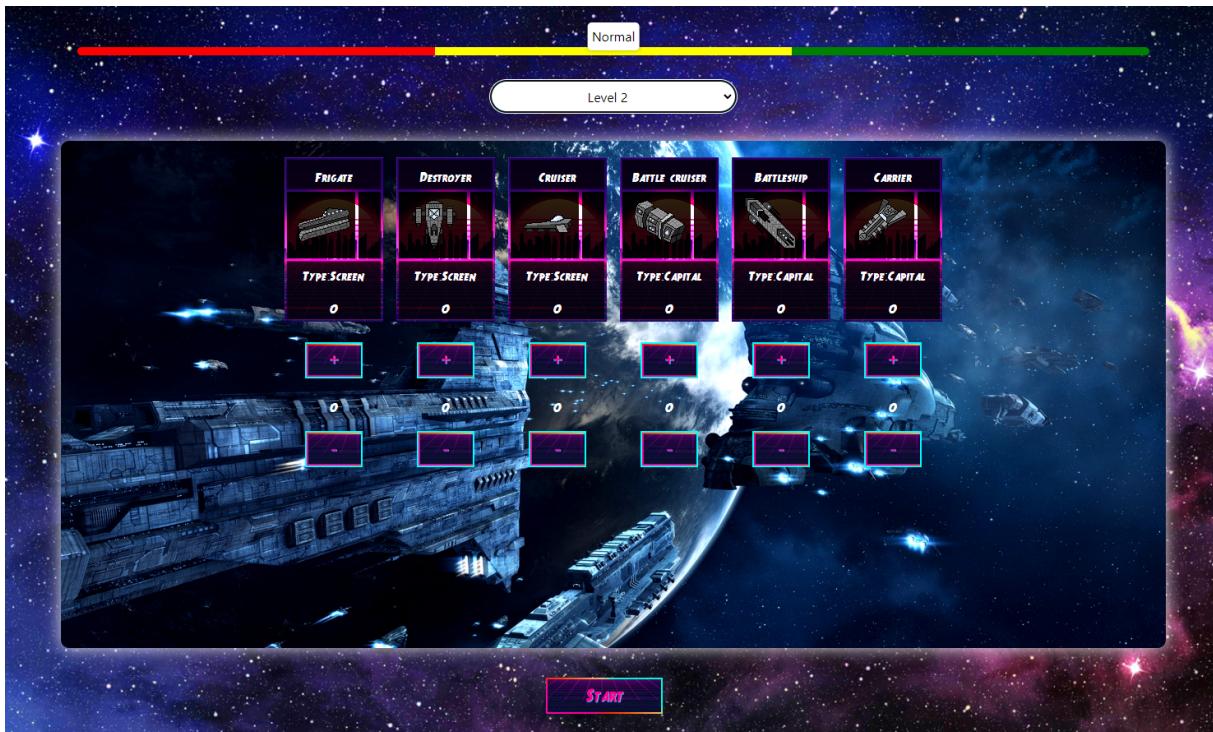
ShipModal: Itt lehet új hajókat létrehozni a flottához.

Battle:

A BattlePage egy olyan oldal ,ahol a felhasználó flottájának összeállítása és bolygók elleni küzdelem indítása lehetséges. Az oldalon található mezők és gombok közül a legfontosabbak:egy lenyíló menü, amelyben a felhasználó választhat a rendelkezésre álló bolygók közül,hat darab hajó kártya, amelyekkel a felhasználó beállíthatja flottájának összetételét és egy Start gomb, amely elindítja a küzdelmet a kiválasztott bolygóval. A számlálók segítségével a felhasználó kiválaszthatja, milyen típusú hajókból szeretné összeállítani a flottáját. Az oldalon a felhasználó a bolygó lenyíló menüből választhat egy célpontot. A küzdelem indításához a felhasználónak a számlálók segítségével össze kell állítania egy flottát. Ha a felhasználó az Start gombra kattint, az alkalmazás elküldi a kiválasztott célpont bolygó azonosítóját és a felhasználó flottáját az API-nak, hogy elindítsa a küzdelmet.Ha a felhasználó hibásan adja meg a flotta összetételét vagy a bolygó, az oldal figyelmeztető üzenetet jelenít meg. Ha a küzdelem indítása sikeres volt, az oldal a küzdelem eredményét kiírja.



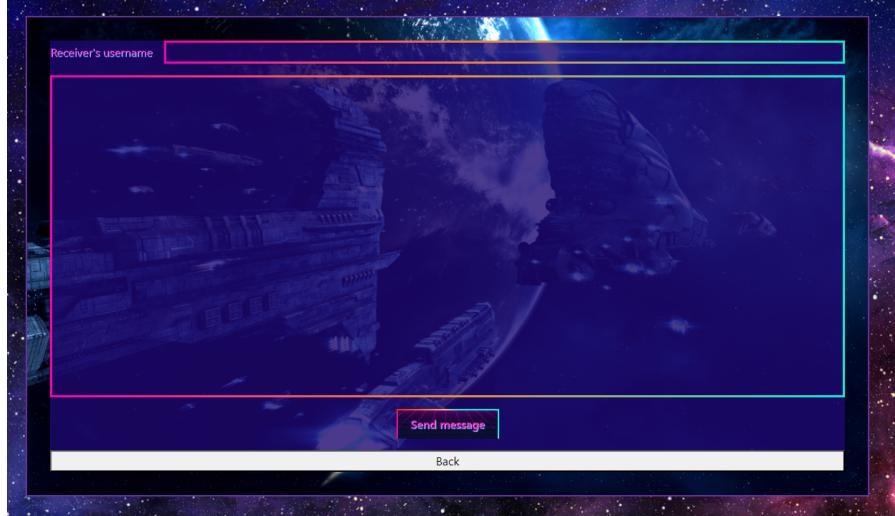
Exploration:



A kód egy React alkalmazás része, amely lehetővé teszi, hogy az egyes játékosok elindítsanak expedíciókat a játékban. Az oldal tartalmazza az összes szükséges funkciót, amelyek az expedíciók kezeléséhez szükségesek. Az oldal használatához az alkalmazás betölti az összes adatot, amely szükséges a játékos hajóinak kezeléséhez, beleértve a hajók típusát és számát is. Az oldal további funkciói közé tartozik az expedíciók nehézségének és a hajók számának beállítása, valamint az expedíciók indítása és végrehajtása. Ezután a felhasználó az oldalon található különböző gombokkal és beviteli mezőkkel beállíthatja az expedíciók nehézségét és a hajók számát, majd elindíthatja az expedíciót. A felhasználó beállíthatja a hajók számát minden típusból külön-külön. Az oldal a hajók számát automatikusan frissíti, amikor a felhasználó megnyomja a Start gombot. Az oldal továbbá lehetővé teszi a felhasználók számára, hogy megtekinthessék az expedíciók közötti minimum időt.

Messages:

Ezen az oldalon a felhasználók üzeneteket küldhetnek egymásnak és megtekinthetik a korábban küldött üzeneteket. Az alkalmazás két fő funkciója az összes üzenet megjelenítése és az új üzenet



létrehozása. Az üzenetek megjelenítéséhez a komponens a MessagesContext-nevű React kontextust használja, amely tartalmazza az összes üzenetet és a pushMessage függvényt, amely lehetővé teszi az új üzenetek hozzáadását az üzenetek listájához. Az összes üzenet megjelenítéséhez a komponens egy AllMessages nevű alkomponenst használ, amely a kontextusban lévő összes üzenetet megjeleníti csevegések formájában. Az egyes csevegések kattintásával az felhasználók megtekinthetik a beszélgetéseket a kiválasztott felhasználóval. Az AllMessages komponens a kontextusban található pushMessage függvényt használja az új üzenetek hozzáadásához a listához. Az új üzenet létrehozásához a komponens egy CreateMessage nevű alkomponenst használ, amely lehetővé teszi a felhasználók számára, hogy új üzenetet küldjenek más felhasználóknak. Az új üzenet elküldése után a felhasználó visszatér az AllMessages oldalra, ahol megtekintheti a csevegést. A Messages komponens használ néhány React hookot, mint például a useState és a useContext hookot, amelyek lehetővé teszik az állapot és a kontextus kezelését a komponensen belül. Emellett az Axios nevű könyvtárat használja az API-hívásokhoz és a SweetAlert nevű könyvtárat a felhasználói értesítésekhez. Az összes üzenet megjelenítése, a beszélgetések megtekintése és az új üzenet létrehozása oldalak közötti navigáláshoz a komponens a React Router nevű könyvtárat használja, ami lehetővé teszi, hogy a fő komponensben az url szerint töltsük be a kívánt alkomponenseket.

FORRÁS

1. Téma: HTML, webcím: <https://hwellkft.hu/marketing-szotar/html>
2. Téma: JWT token, webcím: <https://supertokens.com/blog/what-is-jwt>
3. “2. ábra”: JWT token, webcím: <https://supertokens.com/blog/what-is-jwt> letöltés dátuma: 2023.05.04
4. Téma: Entity Framework, webcím: <https://learn.microsoft.com/en-us/ef/>
5. Téma: ReactJS, webcím: <https://blog.hubspot.com/website/react-js>
6. Téma: MySQL, webcím: <https://www.oracle.com/mysql/what-is-mysql/>