



ZÁRÓDOLGOZAT

Készítették:

Lekner Norbert

Mészáros Balázs

Szigili Edit Mária

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
ISLANDERS
Játék

Lekner Norbert - Szigili Edit - Mészáros Balázs

2022 - 2023

TARTALOM

TARTALOM.....	3
TÉMAVÁLASZTÁS.....	5
A JÁTÉK ALAP KONCEPCIÓJÁNAK BEMUTATÁSA.....	5
Szigetek:.....	6
A játék négy fő része:.....	6
Expedíció: 2. ábra - fő épületek.....	6
Csata:.....	6
Kereskedelem:.....	7
Építés és fejlesztés:.....	7
Az öt alap épület:.....	7
Képességek:.....	8
Alapanyagok.....	8
FELHASZNÁLT TECHNOLÓGIÁK, PROGRAMOZÁSI NYELVEK.....	9
FRONTEND.....	9
HTML:.....	9
CSS:.....	9
React:.....	9
Bootstrap és React-Bootstrap:.....	10
Axios:.....	10
Rxjs:.....	10
React-Cookie:.....	11
Moment:.....	11
React-Simple-Animate:.....	11
React-Hook-Form:.....	12
Use-Sound:.....	12
BACKEND.....	13
Asp.NET Core 6.0 Web API:.....	13
Entity Framework Core:.....	13
MailKit:.....	13
XAMPP:.....	14
MYSQL:.....	14
JWT TOKEN:.....	14
Websocket (signalR):.....	15
NUnit:.....	16
MoQ:.....	16
Lighthouse:.....	16

PROGRAMOZÁSI KÖRNYEZET.....	17
Visual Studio 2022:.....	17
Visual Studio Code:.....	17
ANDROID PORT.....	17
Android studio:.....	17
Capacitor:.....	17
KOMMUNIKÁCIÓS FELÜLETEK.....	18
Trello:.....	18
Github:.....	18
Messenger:.....	18
Discord:.....	18
A FUTTATÁSHOZ SZÜKSÉGES HARDVER KÖVETELMÉNYEK.....	19
Számítógép:.....	19
Mobiltelefon:.....	19
ADATBÁZIS SZERKEZET.....	19
AZ ALKALMAZÁS BEMUTATÁSA.....	21
Bejelentkezés:.....	21
Regisztráció:.....	21
Email cím megerősítése:.....	22
Sziget választás:.....	22
Felhasználói felület:.....	22
Sziget:.....	23
Saját profil:.....	24
Menedzsment:.....	24
Csata:.....	25
Expedíció:.....	26
Piac:.....	27
Útmutató:.....	28
Értesítések:.....	28
Kilépés:.....	28
Zene:.....	28
Mellékletek.....	29
Backend és adatbázis.....	29
Frontend.....	29
Mobil App.....	29
Tesztelési dokumentáció.....	29
Prezentáció.....	29
Repository-kban található.....	29
FORRÁSJEGYZÉK.....	30

TÉMAVÁLASZTÁS

A csoportunk egy szigeten játszódó stratégiai játékot készített webes és mobilos felületre:

Azért esett a választásunk a játékra, mert ebben a rohanó világban, a fárasztó munka mellett az embereknek szükségük van a kikapcsolódásra. Az Interneten rengeteg az értékesítő és informális weboldal, viszont a játék ilyen téren egy kuriózum a webes felületek világában. A játék segít kikapcsolódni, ellazulni, szórakozni. S miközben a többi weboldalt csak szükségleteink kielégítése végett keressük fel, addig ezt az oldalt, vagy alkalmazást a szórakozás kedvéért látogatjuk meg. A játékunkban mindenki saját maga fejlesztheti a saját kis világát, s mindezt olyan ütemben ahogyan ő szeretné. Csatázhat, expedíciókat indíthat, és az épületeiből is gyarapodik a játék folyamán a vagyona. Mindemellett alapanyagai hiánya, vagy többlete esetén cserekereskedelmet is folytathat más szigetekkel (játékosokkal), hogy a fejlődését gyorsítani tudja. A leírásban segítséget nyújtunk, mindazon tudnivalókról, amelyeket érdemes tudni a játékról a fejlődéshez. A játék élményét még fokozzák a mozgó animációk és a szigetspecifikus háttérzene is.

A JÁTÉK ALAP KONCEPCIÓJÁNAK BEMUTATÁSA

A játék lényege, hogy fejleszd a szigetedet. A fejlesztéshez szükséges alapanyagokat más szigetekkel való cserével, csatával, vagy a saját szigeteden indított expedícióval és a szigetedre már telepített épületekből termelődő alapanyagokból tudod beszerezni. A szintlépéshez szükséges tapasztalati pontokat (xp-t) az expedíciókkal és csatákkal tudod összegyűjteni a fejlődéshez. Szintlépésenként kapsz 3 képesség pontot, amelyeket a szigeted alap pontjaihoz tudsz hozzáadni, ezáltal növelheted az erő, ügyesség és intelligencia tulajdonságait a menedzsmentben. Ezeket alul részletezünk. Amennyiben szükséges menet közben a gyors információ, a kérdőjel ikonokra kattintva tudhatsz meg többet.



1. ábra - Islanders logo

Szigetek:

Négy alap sziget van a játékban: Európai, japán, viking és indián. Mindegyik más alap statisztikával rendelkezik és más alapanyag követelményekkel a fejlődéshez.

A játék négy fő része:

- Expedíció a saját szigeten,
- Csata más szigetek ellen,
- Szigetek közötti cserekereskedelem,
- Fő épületek építése és fejlesztése.



2. ábra - fő épületek

Expedíció:

Az expedíciót már a játék elején el tudod indítani. Két expedíció között 1 perc várakozási idő van, ezalatt nem indíthatsz újat. Az expedíció sikerességének eredményét, a begyűjtött alapanyagokkal együtt azonnal megkapod. Három nehézségi fokozat közül választhatsz, természetesen a legkönnyebb adja a legkevesebb anyagot és tapasztalati pontot, a legnehezebb a legtöbbet. A sikerességük kimenetele ugyanígy a nehézségétől függően változik. Az ötös szintig ez az egyetlen lehetőség az xp gyűjtésre a szintlépéshez. Az expedíción kapott nyersanyagot befolyásolja még az intelligencia mennyisége is.

Csata:

Csatázni az ötödik szint elérése után lesz lehetőség. Véletlenszerűen kisorsolt ellenfeleket dob fel a játék és azokat tudod majd megtámadni. Legfeljebb 5 ellenséges sziget közül választhatsz a csaták során. A rendszer a te szintedhez mérten az egy szinttel alacsonyabbaktól a két szinttel magasabbakig adja be az ellenfeleket, hogy ne legyen túl nagy különbség az erőviszonyokban. A csatákat befolyásolják az alábbi tényezők: templom szint, kiképző szint, ügyesség / erő / intelligencia mennyisége. Alul pontosabb információkat kaphatsz róla, valamint az adott menüben (fejlesztésnél, építésnél és tapasztalati pontnál) a kérdőjelre kattintva ott is több információhoz juthatsz.



3. ábra - Expedíció nehéz nehézségi fokozat képe

Kereskedelem:

A kereskedelem a többi szigettel a piac menüpont alatt zajlik. Itt találhatod a játékosok által feladott hirdetéseket. A többi sziget által kirakott portékát az összes hirdetés gombra kattintva tekintheted meg és böngészheted kedved szerint. A hirdetés feladása gombra kattintva tudsz felrakni saját csere ajánlatokat. A feladott hirdetéseidet a saját hirdetések gombra kattintva fogod tudni elérni és törölni, ha meggondolnád magadat. A felkínált portéka a hirdetés feladásakor levonásra kerül a készletedből. Amint valaki e cserét elfogadja, a várt tétel jóvá íródik a te alapanyagaid között. Amennyiben te fogadsz el egy hirdetést, a csere azonnal megtörténik.

Építés és fejlesztés:

Öt alap épületet tudsz lerakni és azokat tudod fejleszteni. Mindegyiknek megvan a sziget fajtájától függően az alapanyag követelménye a megépítéshez és fejlesztéshez. Természetesen több anyag szükséges a második és harmadik szintre való fejlesztéshez, mint az első szint megépítéséhez. Te magad döntöd el, hogy a kijelölt területek közül melyik helyre rakod le az adott épületet és hogy milyen sorrendben. Mindegyik épület termel passzívan alapanyagot és megépítéskor / fejlesztéskor ad egy kevés XP-t.

Az öt alap épület:

Templom:

Passzívan pénzt termel. Csatában plusz sebzést okoz százalékos arányban. Szinttől függően több a sebzés és a termelt pénz mennyisége.

Kiképző:

Passzívan pénzt termel. Csatában a kritikus találat sebzését növeli. Szinttől függően több a sebzés kritikus támadás esetén és a termelt pénz mennyisége.

Fa termelő:

Passzívan fát termel. Szinttől függően emelkedik a kitermelt fa mennyisége.

Kő termelő:

Passzívan követ termel. Szinttől függően emelkedik a termelt kő mennyisége.

Vas termelő:

Passzívan vasat termel. Szinttől függően emelkedik a termelt vas mennyisége.

Képességek:

A képességek elég fontos szerepet töltenek be a szigetek életében. Csatában sokkal nagyobb sebzéshez juthatsz, illetve több anyagot szerezhetsz általuk. Van maximum értékük, így nem rakhatsz minden pontot egy ágra.

Erő:

Növeli a bevitt sebzés mértékét.

Ügyesség:

Növeli a kritikus találat esélyét.

Intelligencia:

Növeli a sikeres expedíció és csata utáni alapanyag és xp mennyiségét.



4. ábra - Pont elosztási menü

Alapanyagok

Három fő alapanyag van a *fa*, *kő*, és a *vas*. Ezek mellett van még *XP* és *érme*. Az érme, fa, kő és vas szükségesek az épületek megépítéséhez és fejlesztéséhez, az XP pedig a sziget össz. szintjének növeléséhez, amely a plusz pontokat adja a képességekhez a fejlődésben.



5. ábra - Az alkalmazás kinézete belépés után (viking sziget választással)

FELHASZNÁLT TECHNOLÓGIÁK, PROGRAMOZÁSI NYELVEK

FRONTEND

HTML:

A HTML mozaikszó (HyperText Markup Language) magyarul Hiperszöveges Jelölőnyelv, leíró nyelv, kódnyelv, a weboldalak készítéséhez fejlesztették. (A jelölő nyelvek, mesterséges nyelvek, amelyeket azért hozták létre, hogy segítségükkel, bizonyos jelölés rendszerrel láthassuk el a szövegeket, képeket, a célunktól függően.) Egy HTML kód egyértelműen meghatározza a weboldal felépítését, kinézetét. Azaz a HTML nyelven megírt kódot a webböngésző értelmezi, majd a kód alapján megjeleníti a már általunk megszokott oldalakat.

CSS:

CSS rövidítés az angol Cascading Style Sheets kifejezésből származik, és az egyik legfontosabb technológia a weboldalak kialakításában. A CSS-t arra használják, hogy megváltoztassák az HTML dokumentumok kinézetét és stílusát, például a betűméretet, a színét, a margókat, a kereteket, a háttérképet és sok mást. A CSS-ben stíluslapokat használunk, amelyek az oldalak megjelenését definiálják. A stíluslapokat külön fájlokban tároljuk, amelyeket a HTML dokumentumokhoz hivatkozunk.

React:

React egy nyílt forráskódú, javascript alapú keretrendszer, amelyet a Facebook fejlesztett ki. A React használatával könnyedén építhetünk újra felhasználható UI komponenseket, amelyek dinamikusan változhatnak a felhasználói interakciók és az alkalmazás állapotának változásai szerint. A React egyik jellemzője az úgynevezett JSX, ami egy olyan nyelv, amely lehetővé teszi a JavaScript és a HTML egyidejű használatát. A JSX-ben definiált elemek a React komponenseknek felelnek meg, amelyek visszatérnek egy felépített DOM elemmel. A React felépítése alapvetően a komponensekre épül, amelyek újra felhasználhatók és összetevői a felhasználói felületnek. Az állapot, a komponensek belső változóit jelöli, amelyek változhatnak a komponens életciklusának során. Az állapot változására történő reakció a React egyik fő erőssége.

Bootstrap és React-Bootstrap:

A Bootstrap egy nyílt forráskódú, front-end webes keretrendszer, amelyet Twitter fejlesztett ki, és célja az volt, hogy egyszerűsítse a weboldalak és webalkalmazások tervezését és fejlesztését. A Bootstrap többek között előre elkészített dizájn elemeket és stíluslapokat tartalmaz, amelyek segítségével könnyen testreszabhatók az oldalak megjelenése és funkcionalitása.

A React-Bootstrap egy Bootstrap alapú front-end keretrendszer, amely speciálisan a React alkalmazásokhoz készült. A React-Bootstrap a Bootstrap funkcióit és stílusait teszi elérhetővé React komponensek formájában, így lehetővé teszi a React komponensek használatát az oldalak fejlesztése során. egyszerűbb tesztelhetőségét.

Axios:

Az Axios egy nyílt forráskódú HTTP kliens JavaScript-ben, amelyet a Node.js és a böngészők közötti HTTP kommunikáció egyszerűsítésére terveztek. Az Axios segítségével egyszerűen küldhetünk aszinkron HTTP kéréseket és kapcsolatot teremthetünk különböző API-kkal és webszerverekkel.

Az Axios nagyon hasznos eszköz az aszinkron HTTP kommunikációhoz, és a React alkalmazásokban gyakran használják az API-khoz történő kapcsolódáshoz és adatok lekérdezéséhez.

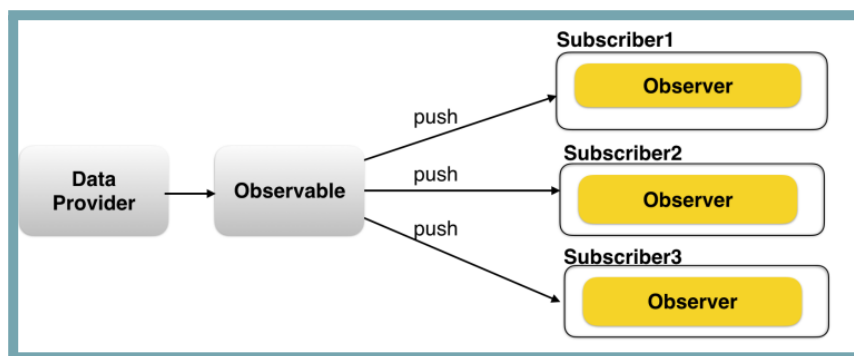
```
axios
.put("https://localhost:7276/api/Auth/ResetPassword", {
  password: password,
  confirmPassword: confirmPassword,
})
.then(() => {
  setUserLoggedOut();
  navigate("");
})
.catch(() => {
  setErrorMessage("Nem sikerült kapcsolódni a szerverhez.");
});
}
```

6. ábra - Axios kódrészlet (put)

Rxjs:

RXJS egy funkcionális reaktív programozási könyvtár, amely lehetővé teszi az aszinkron adatfolyamok kezelését. Az RXJS alapvetően két fő komponensre épül: az Observables és az Operators. Az Observables lehetővé teszik az aszinkron adatfolyamok kezelését, amelyek több értéket adhatnak vissza az időben. Az Observables előnye, hogy könnyen kombinálhatók és manipulálhatók, például szűrhetők, módosíthatók és átalakíthatók más adatformátumokba.

Az Operators olyan funkciók, amelyek manipulálják az Observables-t, például szűrő- és transzformációs műveleteket hajtanak végre az adatfolyamon. Az Operators lehetővé teszik az adatfolyamok kényelmes kezelését és manipulálását. Az RXJS a modern webalkalmazásokban gyakran használt, mivel lehetővé teszi a hatékony és könnyű aszinkron programozást. A React alkalmazásokban is széles körben használják az adatok lekérdezéséhez és kezeléséhez, valamint az alkalmazásállapot kezeléséhez is.



7. ábra - RXJS működése

React-Cookie:

A react-cookie egy könnyűsúlyú könyvtár, amely lehetővé teszi a süti (cookie) kezelést a React alkalmazásokban.

Moment:

A Moment.js egy JavaScript könyvtár, amely lehetővé teszi a dátumok és idők könnyű kezelését, manipulálását és formázását. Azáltal, hogy az objektumok közötti átváltást és manipulációt kényelmesen teszi lehetővé, az alkalmazások fejlesztése során csökkenthető a hibalehetőség.

React-Simple-Animate:

A react-simple-animate egy React könyvtár, amely lehetővé teszi az egyszerű és könnyű animációk készítését. A könyvtár a CSS transition és animation tulajdonságait használja az animációk elkészítéséhez, és lehetővé teszi azoknak a tulajdonságoknak az automatikus generálását, amelyekre szükség van az animációk lejátszásához.



8. ábra - Animációhoz használt sprite-ok

React-Hook-Form:

A React Hook Form egy könnyűsúlyú, teljesítményorientált és kiterjeszthető könyvtár, amely lehetővé teszi az űrlapok egyszerű és hatékony kezelését React alkalmazásokban. A Hook-ok használatával valósítja meg a könyvtár az űrlapok kezelését, és lehetővé teszi az űrlapadatok validálását is.

```
<input
  className="register-input"
  name="password"
  type="password"
  placeholder="Jelszó"
  id="password"
  {...register("password", {
    required: true,
    max: 40,
    min: 8,
    maxLength: 40,
    pattern:
      /^(?=.*\d)(?=.*[a-z])(?=.*[A-Z])(?=.*[a-zA-Z]).{8,40}$/i,
  })}
/>
{errors.password?.type === "pattern" && (
  <span className="reg-error-msg text-center">
    A jelszónak 8 - 40 karakter hosszúnak kell lennie.
  </span>
)}
```

9. ábra - React-hook-form kódrészlet

Use-Sound:

A use-sound egy React hook, amely lehetővé teszi a hangok egyszerű kezelését React alkalmazásokban. A hook a Web Audio API-ra épül, és olyan funkciókat biztosít, mint a hang lejátszása, szüneteltetése, újratekzdése, hangerő és lejátszási sebesség beállítása, valamint a hangfájl betöltése a memóriába.



10. ábra - Zene UI. (bekapcsolt és kikapcsolt állapot)

BACKEND

Asp.NET Core 6.0 Web API:

Az ASP.NET Core 6.0 Web API egy keretrendszer, ami lehetővé teszi a fejlesztők számára az egyszerű és hatékony webes API-k készítését, amelyek skálázhatóak, biztonságosak és platformfüggetlenek. Az API-k adatokat szolgáltatnak más alkalmazások, szolgáltatások vagy felhasználói felületek számára, például a mobil alkalmazások vagy weboldalak számára. Az ASP.NET Core 6.0 Web API használata során az alkalmazás fejlesztése köré épülő MVC (Model-View-Controller) architektúrát használhatjuk, amely lehetővé teszi a fejlesztők számára a kód könnyebb szervezését és karbantartását. Az MVC architektúra lehetővé teszi a kód elkülönítését a különböző fejlesztési feladatokra, például az adatbázis-kezelésre, a felhasználói felületekre és a vezérlőkre. Az alkalmazásokat akár Docker konténerekben is futtathatjuk, amely lehetővé teszi a fejlesztők számára a kód könnyebb és hatékonyabb tesztelését és telepítését. Az ASP.NET Core 6.0 Web API további funkciói közé tartozik a middleware-ek használata, amely lehetővé teszi a fejlesztők számára az alkalmazás különböző szintű beállításait és konfigurációit, az egységes logolás, az egyszerű hibakezelés és a különböző autentikációs és autorizációs funkciók támogatása.

Entity Framework Core:

Az Entity Framework Core (EF Core) egy nyílt forráskódú, könnyű súlyú, keresztplatformos verziója az Entity Framework (EF) objektum-relációs leképező (ORM) keretrendszernek. Az EF Core lehetővé teszi az alkalmazás fejlesztők számára, hogy a .NET alkalmazásaikat objektum-orientált módon építsék fel, majd az EF Core segítségével ezen objektumokat társítsák adatbázis entitásokhoz. Az EF Core segítségével lehetőség van az adatbázisokkal való kommunikációra az LINQ (Language Integrated Query) nyelv használatával, amely lehetővé teszi a fejlesztők számára, hogy az adatbázis lekérdezéseiket és manipulációikat a .NET programozási nyelvhez hasonló módon írják.

Az EF Core könnyen integrálható az ASP.NET Core alkalmazásokba, és számos szolgáltatást és lehetőséget kínál az adatbázis-kezeléssel kapcsolatban, amelyek nagyban segíthetik az alkalmazás fejlesztőket az adatbázisok hatékony és biztonságos kezelésében.

MailKit:

A MailKit egy .NET alapú, nyílt forráskódú email kliens könyvtár, amely lehetővé teszi az SMTP, POP3 és IMAP protokollok használatát a levelezéshez.

XAMPP:

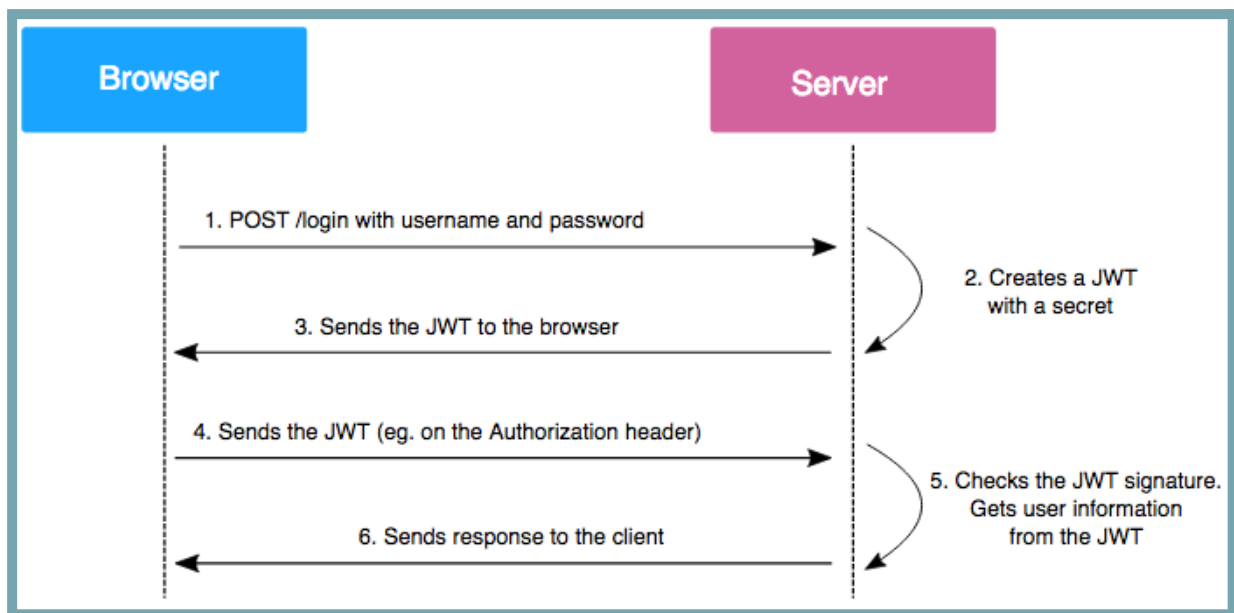
A XAMPP egy ingyenes és nyílt forráskódú szoftvercsomag, amelyet webfejlesztők használnak lokális webkiszolgálók létrehozásához és üzemeltetéséhez.

MYSQL:

A MySQL egy ingyenes és nyílt forrású relációs adatbázis-kezelő rendszer. Az adatok tárolására és lekérdezésére használható, és számos nyelvvel, köztük PHP-vel, Java-val is könnyen integrálható. A MySQL egyik előnye az adatbázis-skálázhatóság, ami azt jelenti, hogy a rendszer képes megbirkózni nagy mennyiségű adattal és sok egyidejű felhasználóval. Az adatok biztonsága érdekében a MySQL beépített biztonsági funkciókkal rendelkezik, például jelszóvédelemmel, szerepkörökkel és hozzáférési jogosultságokkal. Az adatok lekérdezése SQL nyelven keresztül történik, amely az adatbázis-kezelő rendszerek leggyakoribb nyelve.

JWT TOKEN:

A JWT egy nyílt token alapú autentikációs protokoll, amely JSON objektumokat használ az adatok továbbítására az alkalmazások között. A JWT-t általában azonosító és hitelesítő tokenként használják a felhasználói azonosításra és engedélyezésre, de használhatók más típusú adatok továbbítására is. A JWT egy három részből álló karakterlánc, amelyet pontok választanak el egymástól. Az első rész a JWT fejléce (header), amely tartalmazza a token típusát és az algoritmus nevét, amelyet a token aláírásához használnak. A második rész a JWT törzse (payload), amely tartalmazza a hasznos adatokat, mint például az azonosító vagy a felhasználó szerepköre. A harmadik rész a JWT aláírása, amely garantálja, hogy a token valóban a küldőtől származik és nem módosult. A JWT-k nagyon elterjedtek a modern webes alkalmazásokban, mert könnyen használhatók, és nem igényelnek külön adatbázist vagy más háttérrendszert a tároláshoz. A JWT-k általában a szerverről érkező válaszban kapják meg a kliens oldalon, majd a kliens használja a tokent a későbbi kérésekhez. A tokent általában a kérés fejlécében küldik el, és a szerver a tokent ellenőrzi a kért művelet jogosultságának ellenőrzése érdekében. A következő oldal elején található ábra jobban szemlélteti a JWT token működését.



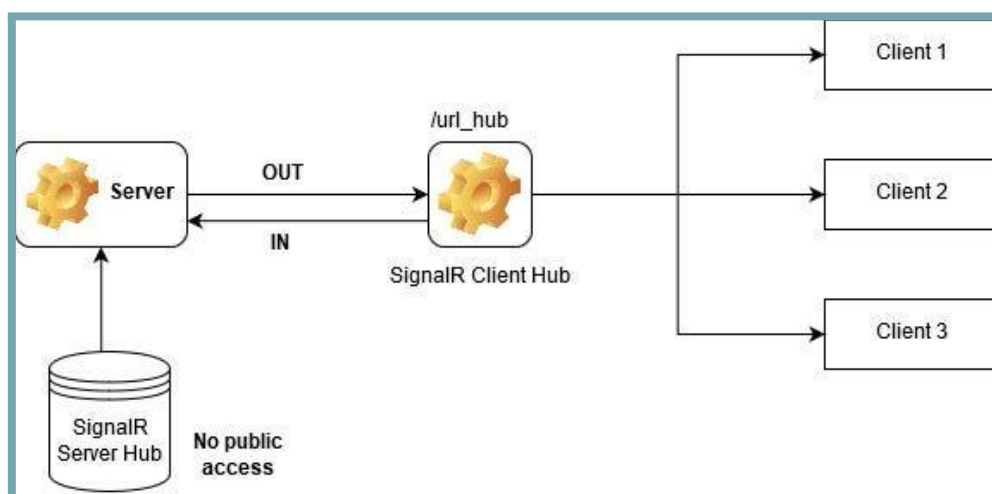
11. ábra - JWT token működése

Websocket (signalR):

A SignalR egy ASP.NET Core alkalmazás, amely lehetővé teszi az alkalmazások számára a valós idejű kommunikációt a kliensek és a szerver között.

A SignalR alapvetően a websocket protokollra épül, amely lehetővé teszi az üzenetek közvetlen küldését a szerver és a kliens között, így valós időben tudják figyelni az adatokat, amelyeket a másik fél küld. Ezt az üzenetküldést "hub" -ok segítségével végzi, amelyek lehetővé teszik a csoportok és a kliensek közötti kommunikáció egyszerű kezelését.

A websocketet sok böngésző támogatja, így a SignalR-en keresztül valós idejű alkalmazások fejlesztése egyszerűbbé válik.



12. ábra - Websocket (SignalR)

NUnit:

Az NUnit egy teszt keretrendszer .NET alkalmazásokhoz, amely segíti a fejlesztőket a hatékony tesztelésben és hibakeresésben. Az NUnit segítségével egységes és szabványos módon írhatunk teszteket, amelyek ellenőrzik az alkalmazásunk különböző részeit és működéseit. A tesztek írásakor különböző attribútumokat használhatunk, például [TestFixture], [Test], [SetUp], [TearDown], amelyek lehetővé teszik a tesztesetek beállítását, futtatását és utólagos tisztítását. Az NUnit lehetőséget nyújt az aszinkron tesztek futtatására is, illetve a tesztek szervezésére és végrehajtására is kínál eszközöket. A tesztek eredményei áttekinthetőek, és azokat a Visual Studio Test Explorer vagy NUnit Test Runner felületén keresztül könnyen lehet megtekinteni és kezelni. Az NUnit hozzájárul az alkalmazás minőségének javításához, a hibák időben történő felderítéséhez és a fejlesztési folyamat hatékonyságának növeléséhez.

MoQ:

Az Moq egy open-source mocking keretrendszer .NET-hez, amely segít tesztesetek írásában és a függőségek kezelésében. A moq-olás segítségével lehetőségünk van arra, hogy olyan objektumokat hozzunk létre, amelyek viselkedése teljes mértékben személyre szabható és vezérelhető, ezáltal tesztelhetővé válnak a függvények, osztályok vagy akár teljes alkalmazások.

Az NUnit tesztelés során a moq-olás használata azt jelenti, hogy létrehozunk egy kitalált objektumot, amely ugyanúgy viselkedik, mint a valódi objektum, de tesztelési célokra használjuk.

Lighthouse:

A Lighthouse egy nyílt forráskódú, automatizált eszköz a webes alkalmazások teljesítményének, minőségének és helyességének javítására.

Egy oldal auditálásakor a Lighthouse tesztek egész sorát futtatja le az oldal ellen, majd jelentést készít arról, hogy az oldal milyen jól teljesített. Innen a sikertelen teszteket indikátorként használhatja arra, hogy mit tehet az alkalmazás javításáért.

PROGRAMOZÁSI KÖRNYEZET

Visual Studio 2022:

A Visual Studio 2022 fejlett szerkesztést, hibakeresést és testreszabást biztosít a mindennapi programozási feladatokhoz. A Visual Studio 2022 egy teljes körű fejlesztői környezet, amelyet a Microsoft készített. Az alkalmazás célja, hogy könnyebbé és hatékonyabbá tegye a Windows, az iOS és az Android alkalmazások fejlesztését, valamint a web- és felhő alapú alkalmazások készítését. Integrált verziókezelést, valamint a kód szerkesztéséhez különböző funkciókat és eszközöket használ.

Visual Studio Code:

A Visual Studio Code (röviden VSCode) egy ingyenes, nyílt forráskódú kódszerkesztő, amelyet a Microsoft készített. Az alkalmazás támogatja a kód szerkesztését és hibakeresését, valamint számos programozási nyelvhez biztosítja a szükséges kiterjesztéseket.

ANDROID PORT

Android studio:

Az Android Studio egy ingyenes, nyílt forráskódú integrált fejlesztői környezet (IDE), amelyet a Google fejleszt az Android operációs rendszerre készülő alkalmazások fejlesztéséhez. Az Android Studio-t a JetBrains IntelliJ IDEA fejlesztői környezetének alapjaira építették, és támogatja az Android platform összes verzióját. Az Android Studio tartalmazza az Android SDK-t, amely tartalmazza az Android operációs rendszerrel kapcsolatos összes API-t, illetve különböző emulátorokat és eszközöket a fejlesztéshez és teszteléshez.

Capacitor:

A Capacitor egy open-source framework, amely lehetővé teszi a fejlesztők számára, hogy natív mobilalkalmazásokat építsenek egyetlen kódbázissal, és a webes technológiákat használják az alkalmazások fejlesztéséhez. A Capacitor lehetővé teszi a fejlesztők számára, hogy hozzáférjenek a natív funkciókhoz és API-khoz, például a kamerához, a helymeghatározáshoz, a Bluetooth-hoz, a hálózathoz és még sok más funkcióhoz.

A Capacitor egy keresztplatformos megoldás, és támogatja az Androidot, az iOS-t és a webet is. A Capacitor támogatja a Cordova pluginokat is, így a Cordova ökoszisztéma részét képezi.

KOMMUNIKÁCIÓS FELÜLETEK

Trello:

A Trello egy webes alkalmazás, amelyet feladatkezelőként használhatunk. Segítségével könnyedén kezelhetjük a projekteket, azokhoz tartozó feladatokat, határidőket, felelősöket, megjegyzéseket, csatolmányokat és sok más információt. A Trello egy ún. Kanban-tábla alapú rendszer, ahol a feladatokat lapokként jelenítik meg, amelyek átlátható módon sorakoznak egymás alatt.

Github:

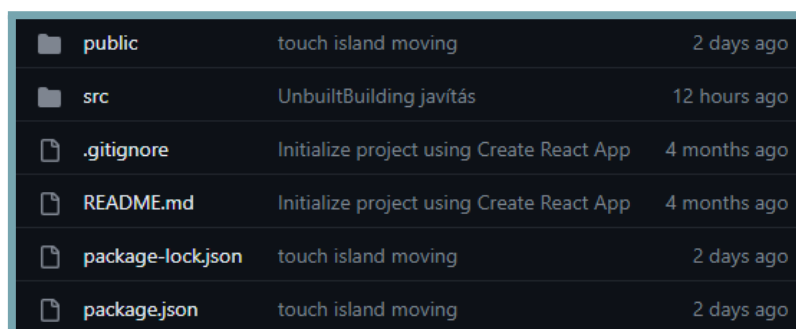
A GitHub egy webes alapú verziókezelő rendszer, amely lehetővé teszi a fejlesztők számára, hogy közösen dolgozzanak ugyanazon a projekt kódján. A GitHub-on a kódokat tárolják és nyomon követik a változásokat. A GitHub a nyílt forráskódú projektek számára ingyenes, és számos szolgáltatást kínál, például verziókezelést, probléma nyomon követést, wiki-t, dokumentációt és együttműködési lehetőséget más fejlesztőkkel.

Messenger:

A legnépszerűbb közösségi oldal, a Facebook 2011. augusztus 9-én adta ki a Facebook Messenger alkalmazást. Üzeneteket lehet vele küldeni, egy helyen kezeli a normál üzeneteket és a chatet, mindezt pedig push notification támogatással.

Discord:

Az egyik legnépszerűbb, ingyenes VOIP (voice over ip) alkalmazás és digitális terjesztési platform. Regisztráció után akár a feltelepített applikációval vagy böngészős megjelenítéssel is használható. Beépített közvetítési szolgáltatással, ami ingyenes bizonyos feltételek mellett.



public	touch island moving	2 days ago
src	UnbuiltBuilding javítás	12 hours ago
.gitignore	Initialize project using Create React App	4 months ago
README.md	Initialize project using Create React App	4 months ago
package-lock.json	touch island moving	2 days ago
package.json	touch island moving	2 days ago

13. ábra - Github frontend repository

A FUTTATÁSHOZ SZÜKSÉGES HARDVER KÖVETELMÉNYEK

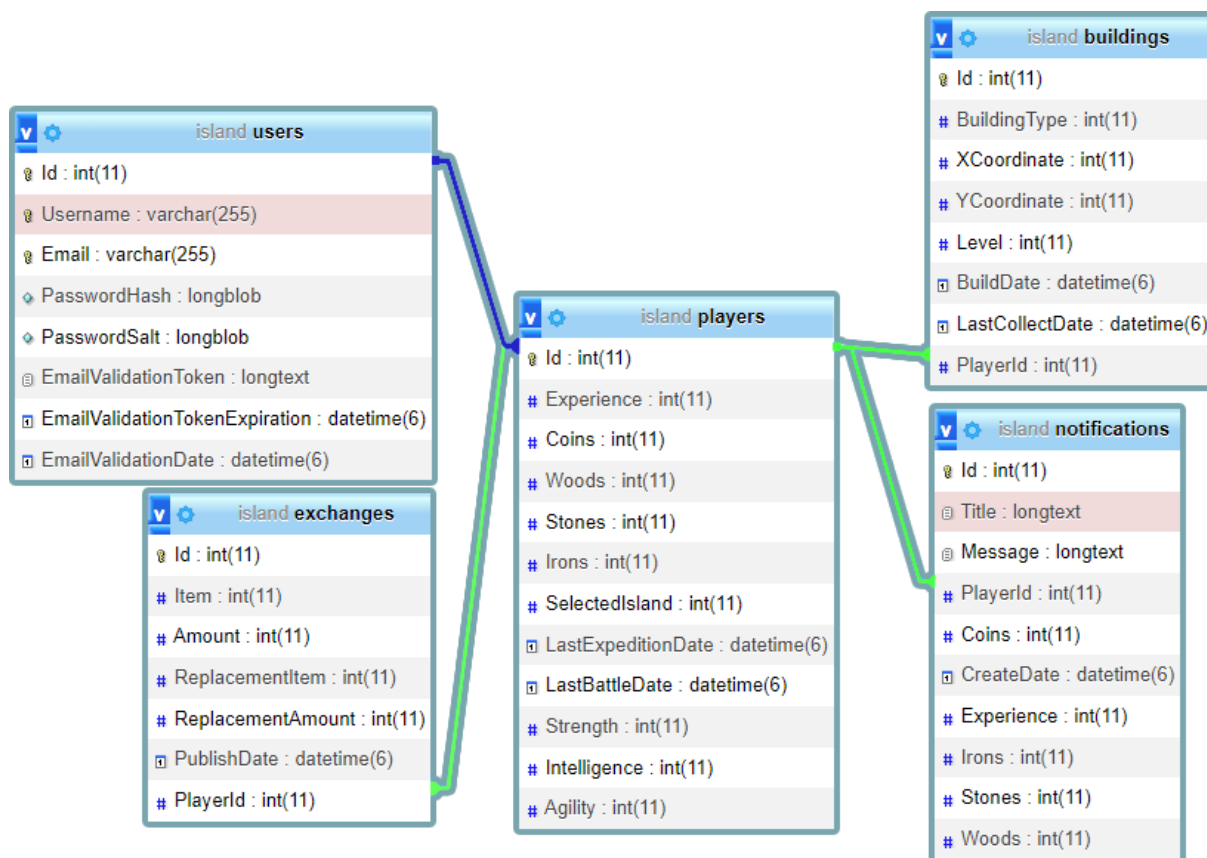
Számítógép:

Telepített böngésző engedélyezett javascripttel, és internet kapcsolat. 1920x1080p felbontás (ajánlott).

Mobiltelefon:

Android 9.0 (pie) vagy frissebb verzió ajánlott.

ADATBÁZIS SZERKEZET



14. ábra - phpMyAdmin kapcsolati nézete

Az általunk használt MySQL relációs adatbázisban tároljuk a felhasználók adatait, amelyek dinamikusan változhatnak. Az adatok 3. normálformára történő átalakítása után öt különböző táblát hoztunk létre. Ezek között különböző típusú kapcsolatok vannak. A "users" és "players" táblák közötti kapcsolat egy az egyhez (1:1), mely azt jelenti, hogy egy

felhasználó csak egy felhasználói fiókkal rendelkezhet és fordítva. A "players" és "buildings" táblák közötti kapcsolat egy a többhöz (1:N) kapcsolat, tehát egy játékos több épületet is építhet. A "players" és "notifications" táblák közötti kapcsolat szintén egy a többhöz (1:N) kapcsolat, egy játékosnak több értesítése is lehet egyszerre. Végül, a "players" és "exchanges" táblák közötti kapcsolat is egy a többhöz (1:N) kapcsolat, ami azt jelenti, hogy egy játékosnak több cseréje is lehet egyszerre a piacon. Az adatbázist az Entity Framework segítségével kezeljük, amelyet model-first megközelítéssel hoztunk létre. Az adatbázis modelljeit definiáltuk a backend alkalmazásban, majd az Entity Framework segítségével automatikusan generáltattuk a táblákat és a köztük lévő kapcsolatokat.

- Users tábla felhasználóra vonatkozó specifikus mezőket tartalmazza, mint email cím, felhasználónév, password salt és hash, email validációs token, illetve a validációs token lejárat ideje és a megerősítés ideje.
- Players táblában kerülnek eltárolásra a sziget adatok. Felhasználó Id, tapasztalati pont, alapanyagok, alap statok, kiválasztott sziget és a legutolsó időpontja a csatának és expedíciónak.
- Buildings táblában az építkezési adatok vannak tárolva. Az épület Id, típus, X és Y koordináta, épület szint, építés dátuma és az utolsó begyűjtés ideje (amit az épületek termeltek).
- Exchanges a kereskedői táblába kerülnek az alapanyagcseréhez szükséges adatok. A hirdetett alapanyag és annak mennyisége, a cserére elvárt alapanyag és annak mennyisége, illetve a kirakás dátuma.
- Notifications táblában az értesítéseket kezeljük. Van mindegyiknek egy címe, üzenete, alapanyag mennyisége és készítési dátuma.

AZ ALKALMAZÁS BEMUTATÁSA

Bejelentkezés:

A jelentkezés felületen látható beviteli mezőkben kell megadni a regisztrált felhasználónevet és jelszavat. Ez indítja el a szerver felé a hitelesítést és jelentkezést. Amennyiben sikeres volt az hitelesítés, a szerver visszaküldi a JWT token-t a kliensnek, amit a böngésző sütibe mentünk el, és a felhasználót átirányítjuk a kezdőoldalra. A token használjuk később az automatikus jelentkezéshez is, amíg ez nem jár le, illetve nem törlik a sütiket, a felhasználót a kliens automatikusan bejelentkezteti. Ha a belépési adatok nem egyeznek az adatbázisban lévő adatokkal, vagy nem létezik a felhasználó, akkor annak megfelelő hibaüzenetet jelenítünk meg a felhasználónak. Illetve a nem megerősített email címmel rendelkező felhasználónak a megfelelő hiba ablakot dobjuk fel, az email megerősítés újraküldési lehetőségével. A belépés gomb alatt található két link: regisztráció és jelszó emlékeztető. Ezek átirányítanak a megfelelő oldalra.

Regisztráció:

A regisztráció oldalon lehet új felhasználót létrehozni. A regisztráció során egy egyedi felhasználónevet, egy még nem regisztrált email címet, valamint egy jelszót és egy megerősítő jelszót kell megadni. Az email címnek valódinak kell lennie, mert később ide küldi majd az alkalmazás a jelszó emlékeztetőt, és a figyelmeztetést, ha megváltozott a fiók jelszava. A megadott jelszó minimum 8 és maximum 40 karakter hosszú lehet, valamint tartalmaznia kell egy kis, egy nagy betűt és egy számot. A felhasználónak hibaüzenet jelenik meg, ha valamelyik kitöltött mező hibás. A sikeres regisztráció után egy email kerül kiküldésre, amiben az email cím megerősítéséhez egy linkre kell kattintani. A link tartalmazza a létrehozott felhasználó email validációs tokenjét, amit a linkre kattintva a kliens elküld a szervernek, majd megtörténik a hitelesítés. A megerősítő tokennek 10 perc a lejárat ideje, ezután nem használható hitelesítésre, újat kell igényelni. A regisztráció során a jelszó titkosítva tárolódik az adatbázisban.

15. ábra - Regisztráció

Email cím megerősítése:

Minden regisztráció után szükséges az email címünk megerősítése, amiről automatikusan küld a rendszer egy levelet a regisztrált email címre. A levél tartalmazza a megerősítéshez szükséges linket, amire kattintva a felhasználó validálni tudja a megadott levelezési címét. Ha az email nem érkezett meg, vagy lejárt az érvényességi ideje, akkor a bejelentkezés menüpont alatt kérhet magának új levelet.

Sziget választás:

Itt választja ki a felhasználó melyik szigeten szeretne játszani. Ez csak első belépésnél jelenik meg. Mindegyik szigetnek más specifikációi vannak (jelentőségéről a dokumentáció elején olvashat). Amint kiválasztottuk a szigetet, az alsó gomb nyomásával elküldjük a szervernek, hogy melyik került kiválasztásra. Majd az elmenti az adatbázisba, az adott számmal. Így belépés után mindig a felhasználói fiókunkhoz rendelt sziget fog betöltődni.

Felhasználói felület:

Belépés után az oldal felhasználói felülete (továbbiakban UI) fogad minket. A háttérben a víz és a sziget elemek vannak (szigetről a továbbiakban olvashat többet). Az UI többi része cssben z-index segítségével mindig a sziget felett marad. Bal oldalt található a menüsor, az oldal tetején az alapanyagok és mennyiségük, végül a jobb alsó sarokban pedig a zene indítás és leállítás gomb. Telefonra is optimalizáltuk, így mobil appra való alakításkor ezzel nem volt teendő.

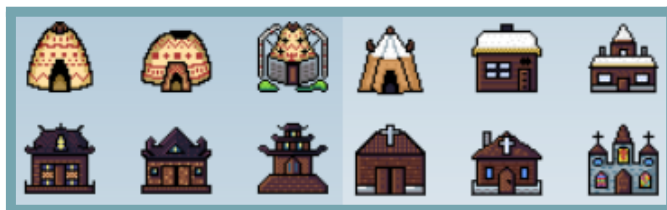
Ha ráhúzzuk az egeret egy menüpontra, akkor megláthatjuk a nevét, mivel nincs külön menüpont írva, hanem alapvetően ikonokkal jelezzük őket. A felső profilképünk az aktuálisan kiválasztott sziget alapján változik, valamint a jobb felső sarkában láthatjuk a felhasználói szintünket. Ez alatt helyezkedik el az tapasztalati pont (továbbiakban XP) bar, ahol jelezzük, hogy jelenleg mennyi XP-nk van éppen és, hogy mennyi szükséges a következő szint meglépéséig.



16. ábra - Mobil UI

Sziget:

Ez a komponens képezi a játék alapját, itt tudja a felhasználó kezelni a regisztrálás után kiválasztott szigetét, valamint ez a komponens töltődik be alapértelmezett kezdőoldalként is. 3 fő funkció érhető el a játékosok számára: itt tudják megépíteni a menedzsment komponensben vásárolt épületeiket (amik később nyersanyagokat termelnek majd nekik), itt tudják a korábban megépített épületeket fejleszteni, valamint az azok által termelt alapanyagokat begyűjteni. 5 féle épület érhető el, emiatt 5 beépíthető koordináta is van szigetenként. A játékos szabadon eldöntheti, hogy melyik koordinátára melyik épületét szeretné lerakni. A komponens képes lenne kezelni több építési területet is (ezek a backenden meghatározhatóak json fájlokban), valamint egyszerre több ugyanolyan típusú épület is megépíthető lenne rajta, de a játék végleges változatában ezeknek a számát korlátoztuk. Találhatóak nem-játékos karakterek is a pályán, amik az előre útvonalként definiált koordinátákon mozognak, dinamikus út kereséssel. Nem interaktálhatóak. Gondolva az androidos változatra és a böngésző kompatibilitásra, csak olyan eszközöket használtunk a fejlesztéshez, amiket biztosan minden mobilos és régebbi böngésző támogat, így nem canvas rajzolja ki a játékkeret, hanem minden templateből lett létrehozva. Emiatt az animációk mennyiségét korlátozni kellett, hogy a gyengébb eszközökön is megfelelő sebességgel fusson a weboldal, így maximum egyszerre csak 6 karakter animáció látható, a lejárt animációkat pedig egy szeméthyűjtő szedi össze. Minden sziget változat saját építési koordinátákkal, nem-játékos karakter útvonalakkal és eltérő textúrákkal rendelkezik, amihez a lerakható épületek kinézete is igazodik. A könnyebb kezelhetőség érdekében a játékos tudja a szigetét mozgatni az egerével, valamint nagyítani és kicsinyíteni azt az egérgörgővel. Ezek a funkciók érintőképernyős eszközön is elérhetőek, értelemszerűen a megfelelő érintés eseményekre reagálva. A nagyítás és a mozgatás is korlátozott, maximum 5x-ös nagyítás támogatott, és a mozgatás során a sziget mögötti háttér nem mehet ki soha a képernyőből. A sziget és az épület adatok csak 1x töltődnek be az alkalmazás futása során, hogy gyorsabb legyen a sziget betöltése a vissza navigálások során.



17. ábra - Épület sprite-ok

Saját profil:

A saját profil tartalmazza felhasználó adatait, a felhasználónevet, email címet, és itt oldható meg a jelszó módosítása is.

A jelszó módosításánál meg kell adni az új jelszót. Az új jelszó megadásakor ugyanazok a paraméterek az irányadók, mint a regisztrációnál. A megadott jelszó minimum 8 és maximum 40 karakter hosszú lehet, valamint tartalmaznia kell egy kis, egy nagy betűt és egy számot. A felhasználónak hibaüzenet jelenik meg, ha valamelyik kitöltött mező hibás. A sikeres jelszó módosításról email értesítést kap a felhasználó.



18. ábra - Saját profil menü

Menedzsment:

A játékosok rendelkeznek képesség pontokkal, amiknek a fejlesztésével előnyre tehetnek szert csata és expedíció közben. Az erő például a sebzés mértékét növeli, az ügyesség a kritikus találat esélyét, az intelligencia pedig sokszorozza a megszerezhető jutalmakat. Minden szintlépés után 3 képességpont kerül jóváírásra a játékosnak, amiket a menedzsment komponensben lehet elosztani. Az épületeket is innen lehet megvásárolni, amik majd a nyersanyag termelésben és a csatában fognak fontos szerepet játszani. A megépítésük csak akkor engedélyezett, ha a játékosnak rendelkezésére áll a megfelelő nyersanyag mennyiség, egyébként az építés gomb inaktív. Minden épület mellett található egy kis kérdőjel ikon, amire kattintva tájékozódni lehet az épületről, és megtekinthető a szükséges alapanyagok mennyisége is. Az épületek legtöbb adata statikus, ezek a backenden vannak tárolva json fájlokban. Az megépítésük és fejlesztésük is idővel jár, ez alatt az épületek helyén egy építést jelző textúra áll, illetve nem termelnek.



19. ábra - Sziget menedzselés menüpont

Csata:

A csata komponens (hasonlóan a sziget komponenshez) mozgatható és nagyítható. A játékos a csatára navigálva maximum 5, a szintjéhez megfelelően sorsolt ellenfelet kap a regisztrált felhasználók közül. Minden ellenfél egy kis szigetként jelenik meg, és a sziget felett egy kis felugró ablak jelzi a játékos felhasználónevét, annak szintjét és életét a saját életünkhöz képest. Csak 5. szinttől érhető el a játékosoknak ez a funkció, és nem is lehet 5. szintet el nem ért szigettel harcolni. A harc akkor indul el, ha kiválasztunk egy ellenfelet és a megerősítő ablakban az indítás gombra kattintunk. A csaták körökre vannak osztva, amit mindig a támadó játékos kezd. Minden körben egy-egy támadási lehetősége van a harcolóknak, amiknek a hatékonysága és ereje a templom szintjétől, a sziget szintjétől, valamint a képesség pontoktól függ. Azonban úgy van kialakítva a harc rendszer, hogy az esélytelenebb ellenfélnek is van lehetősége nyerni. Amikor a backenden lefutott a harc az eredménye megjelenik a csatát indító felhasználónál, illetve mindkét résztvevő valós idejű értesítést a kap a megnyert, vagy elvesztett harcról.



20. ábra - Csata menü (szigetek megjelenítése)

Expedíció:

Az expedíció lényege, hogy a szigeten az alapanyagokhoz viszonylag könnyen, gyorsan hozzájut a játékos, hogy az épületeit minél hamarabb le tudja helyezni, és fejleszteni tudja. Ugyanis a lehelyezett épületeiből ezután kezdődhet csak el a saját alapanyag termelése. Az expedícióknak két végkimenetele lehetsége a sikeres vagy sikertelen expedíció, mely utóbbi természetesen nem hoz alapanyagokat a házhoz. Elindítani az expedíciókat percenként lehet. Három expedíció közül lehet választani:

Könnyű:

A könnyű expedíció indításakor az esetek legnagyobb részében az expedíció sikeres zárul, viszont kevés alapanyag, xp és érme kerül a zsákmányba.

Normál:

A normál expedíció indításakor a kevesebb az esély a sikerre, de a könnyű expedícióhoz képest több alapanyag, xp és érme kerül a zsákmányba.

Nehéz:

A nehéz expedíció esetén jóval kevesebb lesz az esélye a sikeres kimenetelnek, de a megszerzett alapanyagok, xp és arany mennyisége határozottan több lesz.



21. ábra - Expedíció menü és választási lehetőségei

Az expedíció elindításakor a kliens elküldi a szerver felé, hogy melyik erősséget választottuk. A szerveren lefut az expedíció, ekkor az adatbázisban hozzáadja a végeredményt. A szerver értesítést küld a frontendnek, ahol a adatok megjelenítődnek egy ablakban, itt kiírja mi lett az eredmény és mit kaptunk (ha kaptunk), ezeket az adatokat utána az értesítések menüponton belül a felhasználó számára is elérhetővé teszi a későbbi megtekinthetőség végett. Ezután indul el az 1 perc várakozás.

Piac:

A piac menü az egyik fő funkciók egyike az oldalon. Itt lehet a többi szigettel cserélni, saját hirdetést felrakni és kezelni az általunk készített hirdetéseket. A menü indulása után lekéri a szerver az összes feltett piaci hirdetést és kiírja az oldalon őket egymás alá helyezve létrehozás szerint.

Csere:

A csere gombra kattintva elindul a csere szerver felé, ami a saját és a hirdetést feladó személy alapanyagaiból levonja és hozzáadja az adott mennyiséget. Ezután eltűnik a hirdetés és az state változóból is törlődik. Kapunk erről a tranzakcióról egy üzenetet is a websocket-nek köszönhetően. Van lehetőség szűrésre a négy alapanyag szerint, ilyenkor a state változóból kiszűri a megfelelő anyagot. Így könnyebb megtalálni, amit keresünk.

Saját hirdetések kezelése:

Mikor megnyitjuk ezt az almenüt, a state változóba letölti a szerver az adatbázisból az általunk felrakott hirdetéseket, feladás dátumának megfelelő sorrendben. Amennyiben szeretnénk megszüntetni a hirdetést, mert elértük vagy nincs már rá szükségünk, akkor a törlés gombra kattintva töröljük a tranzakciót és eltávolítjuk a hirdetést. Ennek köszönhetően vissza is kapjuk azt a mennyiséget az adott alapanyagból, amit felraktunk cserére.

Hirdetés feladás:

A hirdetés feladása menüpontnál előjön egy form, ami megkérdezi, hogy melyik anyagból szeretnénk cserét indítani és mire szeretnénk cserélni. Mennyiséget is meg kell adni, aminek minimum 1 és maximum 9999-nek kell lennie. Legördülő menüből kell kiválasztani az alapanyagokat és beviteli mezőbe saját magunknak kell beírni, hogy mennyit szeretnénk cserélni. Feladásnál a kliens elküldi a szervernek a megfelelő adatokat és az felrakja a csere adatbázisába a hirdetést.

Útmutató:

Az útmutató a felhasználónak nyújt segítséget a játék megismeréséhez. A felhasználó információkat talál a játékban előforduló elemekről, alapanyagokról, az építkezésről, fejlesztésekről, expedíciókról, csatákról, és a cserekereskedelemről. Bemutatja a játék menetét, a fő részeit, így megismerheti, hogy hogyan tud fejlődni a játék alatt a kiválasztott szigete. Nagy segítség ez annak, aki először lát neki az Islanders játék kalandos világának.

Értesítések:

Az értesítések menüben jelennek meg a játék során kapott üzenetek. Minden expedíció, csata, csere és eladás után, valamint első belépésnél kapunk egy üzenetet. Az üzeneteket az adatbázisban található notifications táblába tárolja a szerver. Mindegyik üzenetnek van egy címe, szövege, alapanyag, xp, coin adatai és dátuma. Mikor megnyitjuk a menüpontot, az alkalmazás lekéri az adatbázisból a játékoshoz tartozó üzeneteket. Lehet ezeket törölni, amiket a state változóból egyből eltüntet a rendszer, így nem kell újratölteni az oldalt. Illetve meg lehet nyitni az üzeneteket, ilyenkor új ablak nyílik, ahol megtekinthetjük a teljes üzenetet. Websocket segítségével folyamatos kapcsolatot tart a szerver és a kliens. Tehát ha valaki megtámad minket, az alkalmazás egy piros felkiáltójeles ikont rak a menüpont ui felé, ezzel jelezve, hogy jött egy új értesítés. Amint megnyitjuk az új értesítés körül egy barna szaggatott szegély található, így tudjuk megkülönböztetni az új vagy nem olvasott üzeneteket. Addig így marad, amíg nem nyitjuk meg a “megnyitás” gomb használatával.

Kilépés:

Kilépés menüre kattintva az alkalmazás törli a sütiben tárolt bejelentkezési adatokat és leállítja a zenét amennyiben fut.

Zene:

Jobb alsó sarokban helyezkedik el. Külön narancssárga effektet kap, ha be van kapcsolva. A legtöbb böngésző szabályai szerint nem lehet elindítani az oldal betöltésénél semmilyen hangot, csak ha a felhasználó indítja el őket. Így addig nem indul el, amíg rá nem kattintunk. A zene folyamatosan újrajátszódik és kilépésnél leáll. A react use-sound npm csomagot használtam erre, mert nagyon egyszerűen megoldható a zene és sima hangok importálás utáni használata. Indításkor meghívja a lejátszás függvényt, ami kikeresi az aktuális szigethez tartozó számot és elindítja.

MELLÉKLETEK

Backend és adatbázis

<https://github.com/LeknerNorbert/Islands>

Frontend

https://github.com/fireball90/island_base

Mobil App

https://github.com/fireball90/islanders_mobilapp

Tesztelési dokumentáció

Backend link: <https://github.com/LeknerNorbert/Islands/tree/master/Dokumentacio>

Frontend link: https://github.com/fireball90/island_base/tree/main/Dokumentacio

Prezentáció

Backend link: <https://github.com/LeknerNorbert/Islands/tree/master/Prezentacio>

Frontend link: https://github.com/fireball90/island_base/tree/main/Prezentacio

Repository-kban található

Ez a dokumentáció és a tesztelési dokumentáció egyaránt megtalálható a frontend és backend repository-kban (PDF és DOCX formátumban). Valamint a Prezentáció mappában megtalálható a magyar és angol verziója a prezentációnak (PDF és PPTX formátumban). A readme file-ban, illetve a repository-ban leírásképp szerepel a localhost-os elindítás leírása. Jelenleg deployolva van a projekt weboldalon, de 60 napig tart csak az ingyenes szerver, amelyet a SharkASP.net segítségével csináltunk. Valamint Google Firebase segítségével lett kirakva maga a weboldal.

FORRÁSJEGYZÉK

1. Téma: Mi az a HTML?, webcím: <https://hwellkft.hu/marketing-szotar/html>
2. Téma: Visual Studio 2022, webcím: <https://learn.microsoft.com/hu-hu/training/modules/visual-studio-intro/>
3. Téma: A messenger kommunikációs felület , webcím: <https://marketingseo.hu/messenger-mint-chat-es-hirdetofelulet/>
4. Téma: React, webcím: <https://react.dev/>
5. Téma: RXJS, webcím: <https://rxjs.dev/>
6. Téma: CapacitorJS, webcím: <https://javascript.plainenglish.io/capacitor-turn-your-web-app-into-a-mobile-app-4d114249e55b>
7. “7. ábra”, webcím: <https://yakovfain.com/2017/08/28/rxjs-essentials-part-1/> letöltés dátuma: 2023.04.20
8. “11. ábra”, webcím: <https://www.vaadata.com/blog/jwt-tokens-and-security-working-principles-and-use-cases/> letöltés dátuma: 2023.04.20
9. “12. ábra”, webcím: <https://blog.ignaciobrasca.com/csharp/signalr/2021/07/25/how-notification-system-signalr.html> letöltés dátuma: 2023.04.20