ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

NEUMANN JÁNOS
INFORMATIKAI KAR

SZAKDOLGOZAT

OE-NIK Hallgató neve: **Lengyel Dániel**

2023 Hallgató törzskönyvi száma: **T/006283/FI12904/N**

Többjátékos repülés szimulátor androidra

OE-NIK	Hallgató neve:	**Lengyel Dániel**
2023	Hallgató törzskönyvi száma:	**T/006283/FI12904/N**

Óbudai Egyetem
Neumann János Informatikai Kar
Szoftvertervezés és -fejlesztés Intézet

SZAKDOLGOZAT FELADATLAP

Hallgató neve: **Lengyel Dániel**
Törzskönyvi száma: T/006283/FI12904/N
Neptun kódja: [redacted]

A dolgozat címe:

Többjátékos repülés szimulátor androidra
Multiplayer flight simulator for android

Intézményi konzulens: Kovács András
Külső konzulens: Fésüs Péter

Beadási határidő: 2022. december 15.

A záróvizsga tárgyai: Számítógép architektúrák
Szoftvertervezés és –fejlesztés specializáció

A feladat

Tervezzen és készítsen egy olyan játékot, ami repülőgépekkel történő repülést szimulál. A repülés fizikai törvényeken alapuljon. Több repülőgépet is legyen lehetőség kipróbálni, illetve többféle módon lehessen ezeket irányítani. A játékosoknak legyen lehetőségük együtt, egy világban repülni.
A program legyen mobil eszközökön futtatható. Ennek megfelelően legyen a felhasználói felület kialakítva, illetve a grafika is legyen optimalizálva, hogy az alacsonyabb teljesítményű készülékkel rendelkező játékosoknak is legyen lehetőségük játszani.
A játékot legyen lehetőség virtuális valóság szemüveggel is kipróbálni. A játék kövesse le a készülék térbeli orientációját, hogy a játékos körbe tudjon nézni a játékbeli térben. Biztosítson olyan irányítási módot, ami akkor is használható, ha a készülék éppen a játékos fején található, aki ezáltal nem fér hozzá a kijelzőhöz.

A dolgozatnak tartalmaznia kell:

- a megoldandó feladat pontos leírását és részletes elemzését,
- a lehetséges megvalósítási módokat és megoldásokat,
- a részletes rendszertervet és annak indoklását,
- a megvalósítás során használt technológiák bemutatását,
- a megvalósítás menetét, a munkafázisok és a tapasztalatok leírását,
- a tesztelési módszereket, tesztelési terveket és a tesztelés eredményeit,
- az elkészült rendszer alkalmazási, és továbbfejlesztési lehetőségeit,
- online a kifejlesztett rendszer forráskódját, illetve a készített dokumentációkat.

ÓBUDAI EGYETEM * Neumann János Informatikai Kar * 1. Budapest

Ph.

.....
Dr. Vámossy Zoltán
intézetigazgató

A szakdolgozat elévülésének határideje: **2024. december 15.**
(OE TVSz 55.§ szerint)

A dolgozatot beadásra alkalmasnak tartom:

.....
külső konzulens

.....
intézményi konzulens

ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

Neumann János Informatikai Kar

HALLGATÓI NYILATKOZAT

Alulírott hallgató kijelentem, hogy a szakdolgozat saját munkám eredménye, a felhasznált szakirodalmat és eszközöket azonosíthatóan közöltem. Az elkészült szakdolgozatomban található eredményeket az egyetem és a feladatot kiíró intézmény saját céljára térítés nélkül felhasználhatja.

Budapest, 2022. 12. 03

Lengyel Dániel
.....

hallgató aláírása

9. melléklet

ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

Neumann János Informatikai Kar

KONZULTÁCIÓS NAPLÓ

Hallgató neve: Lengyel Dániel

Neptun Kód: [redacted]

Tagozat: nappali

Telefon:

Levelezési cím (pl.: lakcím):

[redacted]

Szakdolgozat címe magyarul:

Többjátékos repülés szimulátor androidra

Szakdolgozat címe angolul:

Multiplayer flight simulator for android

Intézményi konzulens: Kovács András

Külső konzulens:

Kérjük, hogy az adatokat nyomtatott nagy betűkkel írja!

Alk.	Dátum	Tartalom	Aláírás
1.	2022. 02. 20	Szakdolgozat téma tervezése	
2.	2022. 03. 18	Irodalomjegyzék áttekintése	
3.	2022. 04. 15	Tervezési fázis kiértékelése	
4.	2022. 05. 10	Prezentációs próba	

A Konzultációs naplót összesen 4 alkalommal, az egyes konzultációk alkalmával kell láttamoztatni bármelyik konzulenssel.

A hallgató a Szakdolgozat I. (BSc) tantárgy követelményét teljesítette, beszámolóra bocsátható.

.....

Intézményi konzulens

Budapest, 2022. 05. 20

9. melléklet

ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

Neumann János Informatikai Kar

KONZULTÁCIÓS NAPLÓ

Hallgató neve: Lengyel Dániel

Neptun Kód: [redacted]

Tagozat: nappali

Telefon:

Levelezési cím (pl.: lakcím):

[redacted]

Szakdolgozat címe magyarul:

Többjátékos repülés szimulátor androidra

Szakdolgozat címe angolul:

Multiplayer flight simulator for android

Intézményi konzulens: Kovács András

Külső konzulens:

Kérjük, hogy az adatokat nyomtatott nagy betűkkel írja!

Alk.	Dátum	Tartalom	Aláírás
1.	2022. 09. 20	Kutatási fázis áttekintése	
2.	2022. 10. 18	Implementáció tervezése	
3.	2022. 11. 15	Tesztelés, eredmények áttekintése	
4.	2022. 11. 28	Prezentációs próba	

A Konzultációs naplót összesen 4 alkalommal, az egyes konzultációk alkalmával kell láttamoztatni bármelyik konzulenssel.

A hallgató a Szakdolgozat II. (BSc) tantárgy követelményét teljesítette, védésre bocsátható.

.....

Intézményi konzulens

Budapest, 2022. 11. 28

TARTALOMJEGYZÉK

1. A PROBLÉMA FONTOSSÁGA, FELVEZETÉSE

A projekt témája a repülőgép szimulátorok. Pontosabban fogalmazva repülés szimulátorok, mivel itt nem magának a repülőgépnek a működése van szimulálva, hanem hogy miként viselkedik a levegőben. A feladat a tolóerő, a felhajtóerő és a légellenállás kiszámolása a repülőgép mozgásának, orientációjának és alaki tulajdonságainak ismeretében, mint például szárnyfelület.

A repülés már kiskorom óta érdekel. Mihelyt androidos készülékre tettem szert, elkezdtem repülés szimulátorokkal játszani. Egy ponton felmerült bennem az igény, hogy más játékosokkal együtt repüljek. Itt azonban akadályba ütköztem. Az egyetlen szimulátor, ami többjátékos szolgáltatást nyújtott, havi feliratkozási díjat szedett azoktól a felhasználóktól, akik ezt igénybe vették. Így akiknek nem állt módjukban fizetni, nem tudták kipróbálni. A projektem erre a problémára nyújt megoldást azáltal, hogy költséghatékony többjátékos szolgáltatást kínál a játékosoknak, akik ezt térítésmentesen igénybe vehetik.

A projekt egy másik motivációs tényezője az, hogy elkezdtem a pilóta képzést (PPL) a Malévnál Dunakeszin, és 11 repült óránál járok. A képzést egyelőre szüneteltetnem kellett. Viszont ebben a projektben, ugyan virtuálisan, de továbbra is repülhetek.

A játékkal a piacot is meg kívánom célozni. Ennek érdekében, a játékélményt tovább növelem a virtuális valóság játékmóddal. Amennyiben a játékos rendelkezik VR szemüveggel (Google Cradboard, és hasonlók), és a készüléke rendelkezik a szükséges szenzorokkal, 3D-ben is játszhat.

2. A LEHETSÉGES MEGKÖZELÍTÉSI MÓDOK ÉS MEGOLDÁSOK ÁTTEKINTÉSE ÉS ELEMZÉSE

A kutatás során számos repülőgép szimulátort megvizsgáltam (mobil alkalmazások). Ezek a következőek:

2.1. X-Plane 9

A szimulátort a *Laminar Research* cég fejlesztette. Egy képernyőkép a 2.1-es ábrán látható. Az első verziót 2010 december 20-án adták ki. [1]

2.1. ábra: X-Plane 9

Ezt a szimulátort próbáltam ki először. A pozitívumai közé lehet sorolni azt, hogy sok különböző repülőgép típust ki lehet próbálni. Azonban ezek egy része fizetéshez kötött, így akiknek nem áll módjukban fizetni, ezeket nem tudják használni. Az irányítás is nagyszerűen lett megoldva. A játék a gyorsulás érzékelő szenzort használja. Ezzel elérve, hogy a készülék, mint botkormány üzemeljen. Ez felettébb finom és precíz irányítást tesz lehetővé. Ha a készüléket balra döntjük, balra csűrünk, ha jobbra, akkor jobbra csűrünk. Ha magunk felé döntjük, akkor felhúzzuk az orrát, előre döntve pedig lenyomjuk. A funkció, ami legjobban megtetszett, az az, hogy a fejlesztők lehetőséget adnak betekinteni a fizika működésébe azáltal, hogy megjeleníti a gépre ható erőket.

A szimulátornak ezen erősségek mellett több negatívuma is van. A leginkább szembetűnő a pilótafülke hiánya. A játékos több nézőpont között váltogathat, de a pilótafülke nézet nincsen ezek között. Az ehhez legközelebb álló az orrnézet. Itt azonban nem jelenik meg a pilótafülke. De műszerek ennek ellenére vannak. A műszerek megtekintéséhez egy külön felületet kell megnyitni. A probléma abból adódik, hogy ez a felhasználói felület a teljes képernyőt elfoglalja, így a játékos nem lát ki a repülőből, amíg a műszereket figyeli.

A repülőgépek külseje szépen ki van dolgozva, de a kormányfelületek nincsenek meganimálva. Emellett a játéktér megvilágítása borongós, ami szomorú hangulatot kölcsönöz.

Számomra a legnagyobb negatívum, mint ahogy arra a bevezetésben is utaltam, a többjátékos mód hiánya. Itt azonban meg kell említeni, hogy van „Dogfight” játékmód, amiben egy számítógép által irányított repülőt kell levadászni. Ez mindenképpen szórakoztató, de még inkább az lenne, ha ezt a másik repülőt is egy játékos irányítaná.

2.2. X-Plane 10

Mint ahogy azt a neve is sugallja, az X-Plane 9 fejlettebb változata. A 2.2-es ábrán látható. Szintén *a Laminar Research* fejlesztette. [2]

2.2. ábra: X-Plane 10

Az itt szereplő megjegyzések a 10.3.2-es verzióra vonatkoznak.

A szimulátor sok mindenben jelent előrelépést az elődjéhez képest. Először is, a játékos már a pilóta szemszögéből is játszhat. Ez esetben egy szépen kidolgozott pilótafülke jelenik meg. Az élethű megjelenéshez az is hozzáad, hogy a fülkében lévő műszerek legtöbbje animált. Azaz nem csak matrica, hanem ténylegesen működő műszer, ami fontos információval látja el a játékost. Ilyen például a műhorizont, a magasságmérő, sebességmérő, variométer. E mellett a kormányfelületek is mozognak a szarvkormány hatására. Ez a szimulátor is betekintést ad a fizika működésébe, az erők kirajzolásával, akár csak elődje.

Az egyik legnagyobb fejlesztés a lokális többjátékos mód implementálása. Ebben a szimulátorban a játékosok már tudnak egymással játszani, amennyiben közös hálózathoz vannak csatlakozva. Ehhez jó kiegészítés az, hogy a repülőgép modellekhez több lehetséges textúra áll rendelkezésre, amik közül a játékos szabadon váltogathat, ezzel egyedivé téve a repülőjét.

Az azonban hátrány, hogy a repülőgép modellek közül a legtöbb fizetős. Az X-Plane 9 esetében is voltak fizetős modellek, de az arányuk jóval kisebb volt. E mellett még az online multiplayer is hiányzik, de a lokális multiplayer nagy előrelépés.

2.3. Infinite Flight

Az *Infinite Flight LLC* által fejlesztett szimulátor. A 2.3-as ábra egy képernyőkép a játékról. Az első verzió 2013 július 7-én jelent meg. [3]

2.3. ábra: Infinite Flight

Az alábbi leírás a 16.13.0-s verzióra vonatkozik.

Egy professzionális benyomást keltő szimulátor, olyan szofisztikált funkcióval, mint például az autó pilóta: sebesség-, irány- és magasságtartás. A széles funkcionalitásnak azonban negatív hatása van a felhasználói felületre nézve, mivel a sok kis gomb zsúfolttá és komplexé teszi.

Ebben a szimulátorban is kapunk pilótafülke nézetet, és egy kidolgozott kabint. A kormányfelületek is animáltak, azonban az X-Plane 10-zel ellentétben a fülkében szereplő műszerek nem működnek, csak díszként szolgáló matricák. A repülőmodellek terén is javít (az X-Plane 10-hez képest). Sokkal több repülő szerepel a szimulátorban, és ezek között nagy arányban találhatóak ingyenesek. Ezek mellett az egyes modellekhez több alternatív textúra létezik, ezzel több lehetőséget adva a játékosnak repülője személyre szabására.

A szimulátor legnagyobb innovációja az online multiplayer, ami lehetővé teszi, hogy a játékosok a világ (szinte) bármely pontjáról együtt tudjanak repülni. Viszont itt jön elő az a probléma, amit a bevezetés részben előre vetítettem. A többjátékos mód egy havidíjas előfizetéses szolgáltatás. Szintén a professzionalitást erősíti, hogy irányítótorony teszi valóságszerűbbé a szerveren történő játékot. Ez azonban egyben negatívum is, hiszen a szimulátorral laikusok is játszhatnak, akiket ugyan érdekel a repülés, de a rádiózási

protokollokat nem ismerik. Így nekik túl komplex lesz az online játék, és a többieket meg összezavarhatják, ha rádiózás nélkül repülnek.

2.4. Flight Pilot

2.4. ábra: Flight Pilot

Ez a 2.4-as ábrán látható, *Fun Games For Free* kiadó által fejlesztett alkalmazás [4] sokkal inkább egy játék, mintsem komoly szimulátor.

Több, játékokra jellemző motívum is megtalálható benne. Az egyik ilyen a küldetések. Repülés közben különféle feladatokat kap a játékos, mint például egy csomag elszállítása az egyik reptérről a másikra. Amennyiben teljesíti ezt a feladatot, játékbeli fizetőeszközt kap, amin újabb repülőgép modelleket vásárolhat. Játékbeli pénzt valós pénzen is lehet vásárolni. Ez egy felhasználóbarátabb megoldás, mint az előző szimulátorok esetében használt, mivel itt nem szükséges valós pénzt költeni az egyes modellek feloldásához. E mellett több játékmód is megtalálható, ami színesebbé teszi a játékot, például repülő versenyzés.

Továbbá az is kiemelendő pozitívum, hogy egyszerű és intuitív felhasználói felülettel rendelkezik. Ennek azonban az a negatív hatása van, hogy a funkcionalitás erősen szűkített a többi szimulátorhoz képest. A kormány mellett egyedül a gázkar, a fékek és a futóművek vezérelhetőek. Sem az oldalkormány, sem a fékszárnyak nem állíthatóak. Belső nézetet sem kapunk, csak a repülőn kívülről lehet szemlélődni.

A legnagyobb negatívum az, hogy a repülés fizikája jelentősen el lett ferdítve. A repülőket például csak nagyon nehezen engedi a hátukra fordulni. Ahogy nő a bedöntési szög, egy erős forgatónyomaték támad, ami vissza próbálja rántani a gépet. E miatt háton repülni huzamosabb ideig szinte lehetetlen. Annak ellenére, hogy a játék tartalmaz kettő kaszkadőr repülőt is, amik így értelmüket vesztik.

2.5. Összefoglaló

Az általam kipróbált szimulátorok legjobb vonásai a következőek.

A felhajtóerőt kirajzoló funkció elnyerte tetszésemet, így implementálni tervezem. Ez azért is hasznos, mert a fizika hibáinak javításakor fontos információkhoz fog juttatni.

A repülő modelleket az X-Plane 10-hez hasonlóan alakítom ki. A kormányfelületek animáltak lesznek, valamint kidolgozott pilótafülkével és működő műszerekkel fognak rendelkezni. A repülők megvásárlása a Flight Pilotban felhasználóbarát, így én is játékpénzért cserébe teszem ezeket megvásárolhatóvá.

A felhasználói felületet a Flight Pilot-hoz hasonlóan intuitívra tervezem, de a funkciók tárháza szélesebb lesz, emiatt középúton lesz a Flight Pilot és az Infinite Flight között a gombok és csúszkák száma terén.

Mind e mellett az online multiplayernek is szükséges helyet kapnia a szimulátorban, hiszen ez a projekt célja. Ezt viszont az Infinite Flighttal ellentétben nem lesz fizetéshez kötve. Ebből adódóan költséghatékony megoldásra lesz majd szükség. A repülők személyre szabását is lehetővé teszem, hogy a játékosok egyedi gépekkel repülhessenek a többjátékos módban.

3. A RÉSZLETES SPECIFIKÁCIÓ LEÍRÁSA

3.1. A repülőgépek

A játékban legyen lehetőség több repülővel is repülni. A VR miatt fontos, hogy a pilótafülke kidolgozott legyen, ténylegesen működő műszerekkel és animált karokkal (gázkar, pedálok, szarvkormány, stb.).

A játékos kívülnézetre is válthat. A repülőgép kormányfelületei, fékszárnyai, futóművei és hajtóművei is animáltak.

A repülőket össze is lehet törni. Ekkor a kormányfelületek, futóművek, légcsavarok elrejtődnek, és a géptörzs kormos - felrobbant textúrát kap.

3.2. A repülés

Az X-Plane szimulátorokhoz hasonlóan a játékos betekintést nyerhet a fizika működésébe, ha bekapcsolja a repülőre ható erők kirajzolását. Kirajzolásra kerül a hajtóművek tolóereje, a szárnyak felhajtóereje, és a légellenállás is.

3.3. UI

Fontos szempont, hogy a játék felhasználói felülete intuitív és letisztult legyen.

Annak érdekében, hogy különböző nyelvű felhasználók is könnyen eligazodjanak, a gombokon és más elemeken elsősorban ikonokat használok szöveg helyett. Például a beállítások gombon egy fogaskerék ikon látható, a játék indítása gombon pedig egy jobbra mutató háromszög.

A letisztultság elvárásának a következménye, hogy néhány irányítási funkció nem kerül implementálásra. Ilyen például a trimmelés, vagy a több hajtóművel rendelkező gépek esetén az egyes hajtóművek független szabályozása. Ezek hatására a felhasználói felület zsúfoltsága többet rontana a játékélményen, mint amennyit a plusz funkciók javítanak.

3.4. Az irányítás

Több irányítás opció is létezik, amelyek közül a játékos választhat:

Gyorsulásmérős irányítás

A magassági kormány és csűrőlapok irányítása a készülék előre-hátra, illetve balra-jobbra döntésével valósul meg.

Minden egyéb a kijelzőről irányítható. Bal oldalt a gázkar, alul az oldalkormány állítható.

Joystick a kijelzőn

Az opció nagyban megegyezik az előzővel. A különbség az, hogy itt a magassági kormány és a csűrőlapok is a kijelzőről irányíthatóak, egy joystick segítségével.

Egykezes kontroller

Az „esperanza" márkájú mobilkészülékhez tervezett VR szemüveg mellé egy ilyen kontroller járt.

Várhatóan több játékos is ilyen kontrollerrel rendelkezik, így fontos, hogy ez támogatva legyen.

Hagyományos kontroller

Nagyban hasonlít az egykezes változathoz. Azonban a joystick tengelyei fel vannak cserélve, és több gomb foglal rajta helyet. Ha valaki ilyennel rendelkezik, az is tud játszani.

Billentyűzet

Elsősorban teszteléskor hasznos, de elképzelhető, hogy egyes játékosok emulátoron játszanak, így hozzáférnek egy billentyűzethez.

Kiegészítő androidos távirányító alkalmazás

A VR élmény fokozására egy finom irányításra lehetőséget adó távirányító alkalmazást fejlesztek. Ez lehetővé teszi, hogy a játék egy másik androidos eszközről legyen irányítva. Ez azoknak is nagy segítség, akik nem rendelkeznek semmilyen játék kontrollerrel, de szeretnének 3D-ben játszani.

3.5. Többjátékos lehetőségek

A játék központi eleme a többjátékos mód. Ennek kettő fajtája van.

3.5.1. Lokális multiplayer

Ezt a módot választva a játékosok a lokális hálózaton keresztül játszhatnak. Ennél a módnál szükség van egy hosztra, akire a többi játékos csatlakozni tud. Fontos, hogy a játék ne álljon meg, ha a hoszt elhagyja a játékot. Ezért hoszt migrációt is implementálni kell.

3.5.2. Online multiplayer

A játékosok az interneten keresztül a világ (szinte) bármely pontjáról játszhatnak, ezt az opciót választva. A többjátékos mód kialakításának egyik legfontosabb szempontja, hogy költség hatékony legyen.

Ezen belül is több játékmód van:

Free Flight

Nagyban hasonlít az egyjátékos módhoz. A játékos kötetlenül, szabadon repkedhet. Itt azonban még pénzérmék is lebegnek, amiket a játékos összeszedhet. Ezzel a játékbeli pénzzel új repülőket vásárolhat.

Star Collector

Ez egy csapatos kompetitív játékmód. A lényege a következő:

- A két csapatnak külön-külön van egy reptere.
- A pályán csillagok lebegnek.
- Ezeket a játékosoknak fel kell szedniük.
- Majd vissza kell vinniük a csapat repterére.
- Az a csapat nyer, amelyik több csillagot juttat el a bázisára.

A játék időtartama 20 perc.

A nyertes csapat minden tagja 2 ezüstérmét nyer. E mellett mindenki a saját teljesítménye szerint is díjazásra kerül. Minden megmenekített csillag után 1 bronzérmét kap a játékos.

3.6. A gépek egyedivé tétele

Minden repülő burkolatát két anyagminta határozza meg. Annak érdekében, hogy a játékosok megkülönböztethessék magukat, lehetőséget kell nekik adni arra, hogy megváltoztassák ezeket a matériákat egy szerkesztő felület segítségével. Tetszőleges számú matériát létrehozhatnak, amiket hozzárendelhetnek a repülőkhöz. Ezek az egyedi matériák megjelennek a térképeken és a lobby-ban is.

3.7. Biztonsági mentések

A játék adatait ki lehet menteni egy fájlba, amit a játékos egy másik készüléken, vagy a szimulátor újra telepítése esetén visszaállíthat.

Így a játékos biztonságban tudhatja a játékba belefektetett erőforrásokat (idő, energia, pénz). Így végső soron nagyobb bizalommal vásárol játékbeli fizetőeszközt, tudván, hogy nem veszik el.

Ezzel a lehetőséggel azonban vissza is lehet élni. A csalást megelőzendő, a mentést módosítás-biztossá kell tenni. A cél az, hogy ne lehessen a fájlt észrevétel nélkül módosítani.

4. A MEGOLDÁSI MÓDSZER KIVÁLASZTÁSA, A VÁLASZTÁS INDOKLÁSA

4.1. Célplatform

A projektemmel az androidos készülékeket célzom meg. Ennek első oka a mobil eszközök kéznél levősége. Az a célom, hogy minél többen használják a játékomat egy időben. Ezt úgy tervezem elérni, hogy az alkalmazást könnyen elérhetővé teszem. Erre kínálnak megoldást az okostelefonok, amik mindig nálunk vannak, egyfolytában bekapcsolva. Ha valaki, akár csak öt percre is olyan szituációba kerül, hogy valahol várakoznia kell, néhány pillanat alatt már játszhat is.

Ugyancsak nagyszerű lehetőség rejlik a telefonokban lévő szenzorokban, különösen a gyorsulásmérőben. Mint ahogy arról már a konkurencia értékelésénél is szó esett, ez felhasználható a botkormány leutánzására. Ha bedöntjük a készüléket, a játék úgy veszi, hogy a botkormányt mozdítottuk a bedöntés irányába. Ezzel nagyon finoman és precízen tudjuk irányítani a repülőgépet, anélkül, hogy külön be kellene szereznünk egy botkormányt.

Ezen tulajdonságok természetesen nem csak az androidos készülékekre igazak, hanem például az IOS-t futtató eszközökre is. Az android mellett való döntésnek első sorban pénzügyi oka van. A játék egyik fő célja, hogy térítésmentes többjátékos szolgáltatást nyújtson. Emiatt fontos a költségeket minimalizálni, különben ez a cél hosszú távon fenntarthatatlan lenne. A költségekhez a szoftver áruházak (Google Play, App Store) díjai is hozzáadódnak. Egy Google Play fejlesztői fiók regisztrálásához $25-os egyszeri díj befizetése szükséges [5]. Az App Store-nak azonban éves költsége van, méghozzá $99 [6]. E miatt döntöttem első sorban az android mellett. A játékmotor viszont, amivel a projektet fejlesztem lehetőséget ad IOS-re való publikálásra is. Így, ha befut a játék, semmi akadálya annak, hogy az App Store-on is megjelenhessen, de erről majd a fejlesztői környezetben részletesebben lesz szó.

4.2. Fejlesztői környezet

Játékmotor

A szimulátor fejlesztéséhez játékmotort használok, mert így felgyorsul a fejlesztési folyamat. Két lehetséges játékmotor az Unreal engine és Unity. A választásom a Unity engine-re esett. A szempontokat a következő táblázat tartalmazza (4.1. táblázat):

Játékmotor / Szempont	Unreal Engine	Unity
Ingyenes	**Igen**, 5% jogdíjjal	**Igen**
Platformfüggetlen	**Igen**	**Igen**
Intuitív felhasználó felület	Nem	**Igen**
A magas szintű C# nyelvet támogatja	Nem	**Igen**
Alkalmazáson belüli vásárlások	**Igen**	**Igen**
Hirdetések megjelenítése	**Igen**	**Igen**

4.1. táblázat: Az Unreal Engine és a Unity összehasonlítása

A platformfüggetlenség azért szerepel a szempontok között, mert ugyan a cél platform android, de teszteléskor Windows-os buildekre is szükség van, például a többjátékos mód gyors ellenőrzésére. A későbbiekben pedig, ha a szimulátor androidon sikert arat, fel fog merülni az igény, hogy a játék IOS-re is kiadásra kerüljön. A Unity lehetőséget ad arra, hogy könnyen válthassunk ezen platformok között.

A magas szintű C# nyelv szintén a fejlesztés felgyorsításához fontos, mivel a memória kezelést nem kell külön implementálni. A piacot megcélzó projektben szükség van alkalmazáson belüli vásárlásokra és hirdetésekre, így ezeknek a támogatása is szempont.

Android Studio

A projekt android specifikus részeit Android Studio-val fejlesztem, mivel ez a hivatalos android fejlesztő környezet [7]. Továbbá egy távirányító alkalmazást is készítek, aminek segítségével egy másik androidos eszközről lehet majd a repülőgépet irányítani. Ez is Android Studio-ban lesz fejlesztve.

Blender

A játékhoz több háromdimenziós modellt is kell készíteni. A terepet, a hangárokat, és természetesen a repülőgépeket. A költséghatékonyság érdekében az ingyenes Blender modellező programmal készítem el ezeket.

4.3. További aspektusok

A projektnek több lényeges aspektusa van, így nem csak egy megoldási módszer szükséges, hanem aspektusonként (legalább) egy-egy.

Irányítás

Az irányításnak több lehetséges megvalósítása is van, ezek közül azonban nem kell pontosan egyet kiválasztani. Akár az összes ilyen megoldás megvalósítható, az egyes felhasználókra bízva azt, hogy ki melyiket fogja használni.

a. Gyorsulásmérős irányítás

Ez az a megvalósítás, amelyben a telefon gyorsulásmérője által generált bemenetet értelmezi a játék a botkormány mozgatásaként. A magassági kormány és csűrőlapok irányítása a készülék előre-hátra, illetve balra-jobbra döntésével valósul meg. Lehetőség van a kalibrálásra is. Ehhez a felhasználó a telefont egy neki természetes pozícióban kell tartania és megnyomni az újra kalibrálás gombot. Ennek hatására a program megjegyzi a telefon jelenlegi gyorsulását, és ezt veszi viszonyítási alapul.

Minden egyéb, a kijelzőn megjelenő elemek segítségével irányítható. Ilyen például a gázkar, amit egy csúszkával lehet állítani, továbbá a fékszárnyak, az oldalkormány, a fékek, és a futóművek behúzása, illetve kiengedése.

b. Joystick a kijelzőn

Ez az opció abban különbözik az előzőtől, hogy itt a magassági kormány és a csűrőlapok is a kijelzőről irányíthatóak, egy joystick segítségével. E mellett néhány gomb áthelyezésre kerül, hogy a joystick elférjen.

c. Játék kontroller

Ez az opció első sorban a VR szemüveget használó játékosok számára fontos. Ekkor a készülék a felhasználó fején található, bezárva egy dobozba, ami jelentősen megnehezíti a képernyőhöz való hozzáférést. Így a kijelzőt használó irányítási módok ez esetben nem jöhetnek szóba.

Ilyen telefon kiegészítő VR szemüvegekhez gyakran adnak egy kontrollert is. Az én általam használt szemüveghez járt egy. Ezt az opciót választva ilyen kontrollerekkel is irányítható a játék.

d. Billentyűzet

A tesztelés első sorban Windows rendszeren történik. Annak érdekében, hogy a munkámat megkönnyítsem, az egyes funkciókat billentyűkhöz rendeltem, mint például a nézőpont váltás, újrakezdés és a szünetelés. Ez a kezdetleges alkalmazásban nem is

szerepelt, mint opció, de a felhasználók visszajelzései alapján erre a megoldásra is szükség volt, így már mindenki számára elérhető.

e. Alkalmazás

A kontrollernek, amivel kezdetben teszteltem a programot több hátránya is van. Az első az, hogy kevés gomb kap rajta helyet. Nem jut minden funkcióhoz külön gomb, amelyek így nem is elérhetőek. Továbbá a joysticknek is jelentős holtjátéka van, ami lehetetlenné teszi a finom irányítást.

Ennek megoldására egy távirányító alkalmazást fejlesztek, aminek segítségével lokális hálózaton keresztül lehet irányítani a játékot egy másik androidos készülékről. Az alkalmazás az irányításhoz használt eszköz gyorsulásmérő adatait továbbítja a játéknak, ami ezt úgy kezeli, akárcsak a gyorsulásmérős irányítás esetén. E mellett gázkar, fékszárny és oldalkormány csúszka is helyet kap a kijelzőn, illetve a különböző funkciók gombjai (fékek, futóművek, nézet váltás, kalibrálás stb.). Ezek az elemek kellően nagyok, hogy odafigyelés nélkül is könnyen eltalálhatóak legyenek.

Virtuális valóság

A virtuális valóság implementálásához a legkézenfekvőbb megoldás a Google által nyújtott Cardboard SDK használata, ami kifejezetten Unity projektekhez készült. Ezzel azonban az a probléma adódik, hogy az alkalmazás futtatásához szükséges android API szintet megemeli, így kevesebb készülékkel lesz kompatibilis. Továbbá a csomag mérete is jelentős, ami lényegesen megnövelné a játék méretét is. Így amellett döntöttem, hogy a készülék mozgásának lekövetését magam implementálom.

Online multiplayer

A többjátékos mód megvalósítható felhő alapon, dedikált szerver bérlésével. A játékélmény szempontjából ez lenne a megfelelő út, mert gyors adattovábbítást tesz lehetővé a kliensek között.

A hátránya abból adódik, hogy egy ilyen szerver meglehetősen költséges. Az Amazon EC2 szolgáltatását vizsgáltam. A dolgomat nagyban segítette, hogy létezik egy árkalkulátor, ami megközelítőleg megmondja, hogy a megadott számítási kapacitással, memóriával, háttértárral mennyibe kerül a szolgáltatás. Igen szerény beállítások mellett – 2 vCPU, 2 GB memória és 10 GB SSD tárhely esetén – a havi költség $23.05 [8]. Ez megnehezítené a projekt küldetését, hogy díjmentes multiplayer-t nyújtson. Emiatt szükséges egy másik, költséghatékonyabb megoldást találni.

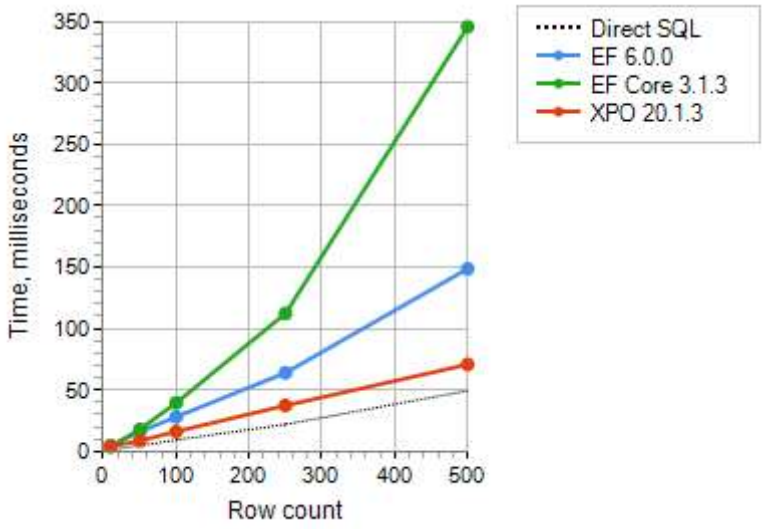
A következő megoldás fogalmazódott meg a fejemben. Legyen egy központi üzenőfal, mint valami fórum, amit minden kliens ismer. Erre az üzenőfalra mindenki felírja a többiekkel megosztani kívánt adatokat, majd elolvassa a többiek által megosztott információkat. Ebben a megoldásban az a lehetőség rejlik, hogy az üzenőfalat nyújtó szerver meglehetősen passzív tud lenni. Csak ráírnak és leolvasnak róla üzeneteket. Az

állapottartás sem a szerver feladata, hanem a klienseké. Ennek hatására jelentősen leegyszerűsödik a szerver, olyannyira, hogy az üzenőfal szerepét egy egyszerű relációs adatbázis is be tudja tölteni. Ennek a megoldásnak természetesen hátránya is van, hiszen az adatátvitel lényegesen lassabb lesz, és a kliensek bonyolult állapottartó logikát igényelnek, cserébe viszont jelentősen olcsóbb. Jelenleg a Cweb.hu tárhely szolgáltatását veszem igénybe, amihez webszerver és adatbázisszerver is jár. Az előfizetésem dollárba átszámolva körülbelül $21 évente. Tehát annyiba kerül egy évre, mint az AWS egy hónapban.

Adatbázis elérés

A többjátékos módot tehát egy központi adatbázissal oldom meg. Az a kérdés, hogy hogyan fog zajlani az adatbázis elérése. A két legfontosabb szempont a lekérdezések sebessége és az adatbázis biztonsága.

Először az ADO.NET Entity Framework kerül szóba, amit az egyetemi tanulmányok folyamán megismertem. Az előnye az, hogy az adatbázis táblákból C# objektumokat állít elő és ezzel sokkal kényelmesebb kezelni az adatbázist, szebb és karbantarthatóbb lesz a kód. A hátránya azonban a teljesítmény. Az Entity Framework (EF) teljesítmény szempontjából alulmarad a közvetlen SQL lekérdezésekhez képest, lásd 4.1. ábra.



4.1. ábra: Az UPDATE művelet időigénye a rekordok számának függvényében. Forrás: https://github.com/DevExpress/XPO/tree/master/Benchmarks

Mivel játékok esetén a sebesség kulcsfontosságú, a közvetlen SQL lekérdezések használata adódik, mint legjobb út. Kezdetben ezt a megoldást használtam. A sebességgel nem is volt gond, a lekérdezések gyorsan lefutottak, de nemsokára elkezdtek felmerülni problémák.

Az egyik nap, minden előjel nélkül elérhetetlenné vált az adatbázis a program számára. Már történt korábban, hogy átmeneti szerverkimaradás volt néhány órára, de ez már több napja tartott. Beléptem a webtárhely fiókomba, hogy megvizsgáljam mi történik, és

meglepődve tapasztaltam, hogy az adatbázis hiba nélkül működik, és a Cweb által nyújtott phpMyAdmin eszközzel el is tudtam érni. Ekkorra már kísérleteztem azzal, hogy PHP oldalakra kiszervezem az adatbázis elérést (egy másik játékhoz), amik szintén gond nélkül működtek. Ekkor egyértelművé vált, hogy az adatbázis csak külső hálózatokról nem volt közvetlenül elérhető, mert a tűzfal blokkolta a forgalmat.

A másik probléma a jelentős biztonsági kockázat. Ahhoz, hogy a kliens csatlakozhasson az adatbázishoz, meg kell adni egy úgynevezett „connection string”-et, ami tartalmazza az összes szükséges adatot, beleértve az adatbázis IP címét, a felhasználónevet és jelszót. A programkód továbbá SQL lekérdezéseket fog tartalmazni, melyekből könnyen kiolvashatóak, hogy milyen táblák vannak az adatbázisban, milyen oszlopokkal. Ez utóbbi önmagában nem jelent kockázatot, de a felhasználónév és jelszó birtokában a struktúrát ismerve bárki könnyedén bele tud kontárkodni az adatbázisba. Rosszindulatú személy úgy tudná módosítani a rekordokat, hogy az a klienseken hibát okozzon, így lehetetlenné téve az online többjátékos módot. Röviden összefoglalva, a probléma az, hogy nem lehet kikényszeríteni az adatbázis rendeltetésszerű használatát, és ez a visszaélés kockázatát hordozza magában.

Mint azt előbb írtam, már kísérleteztem azzal, hogy az adatbázis kezelés PHP végpontokon történjen, amik a tárhely mellé járó webszerveren futnak, és a kliensek csak ezeken keresztül tudjanak adatokat cserélni. Ez biztonsági szempontból már megfelelő. A kliensek HTTP kéréseket intéznek a PHP szkriptek felé, GET paraméterekben átadva az SQL lekérdezéshez szükséges paramétereket. Ezeket a paramétereket ellenőrzi a PHP kód, hogy nincsen-e bele injektált SQL kód. Ezt követően összeállítja a tényleges SQL lekérdezést, lefuttatja, az eredményt pedig visszaadja a kliens felé. Azért nyújt biztonságot ez a megoldás, mert a kliensnek nincsen beleszólása abba, hogy milyen logikát futtat a PHP végpont, így kikényszeríthető az adatbázis helyes használata. Továbbá a szerver dönthet úgy is, hogy nem szolgálja ki a klienst. Meg lehet vonni a szolgáltatást azon felhasználóktól, akiktől rosszindulatú tevékenységet észlel a rendszer, így nagyobb kapacitás jut a tisztességes játékosokra. Mindemellett a klienseknek nem kell tartalmazniuk az adatbázishoz használt felhasználónevet és jelszót, amelyek segítséget nyújtanának sérülékenységek keresésében.

A megoldás hátránya az elkerülhetetlen sebességromlás, mert így nem csak egy SQL lekérést kell futtatni, hanem előbb egy HTTP kérést is. Ennek ellenére ezt az utat választom, mert az előnyei messze felülmúlják a hátrányát, és mert a tapasztalatok azt mutatják, hogy nincs (a felhasználó által) érezékelhető teljesítményromlás ezt a kiszervezett adatbázis kezelést alkalmazva.

Grafika

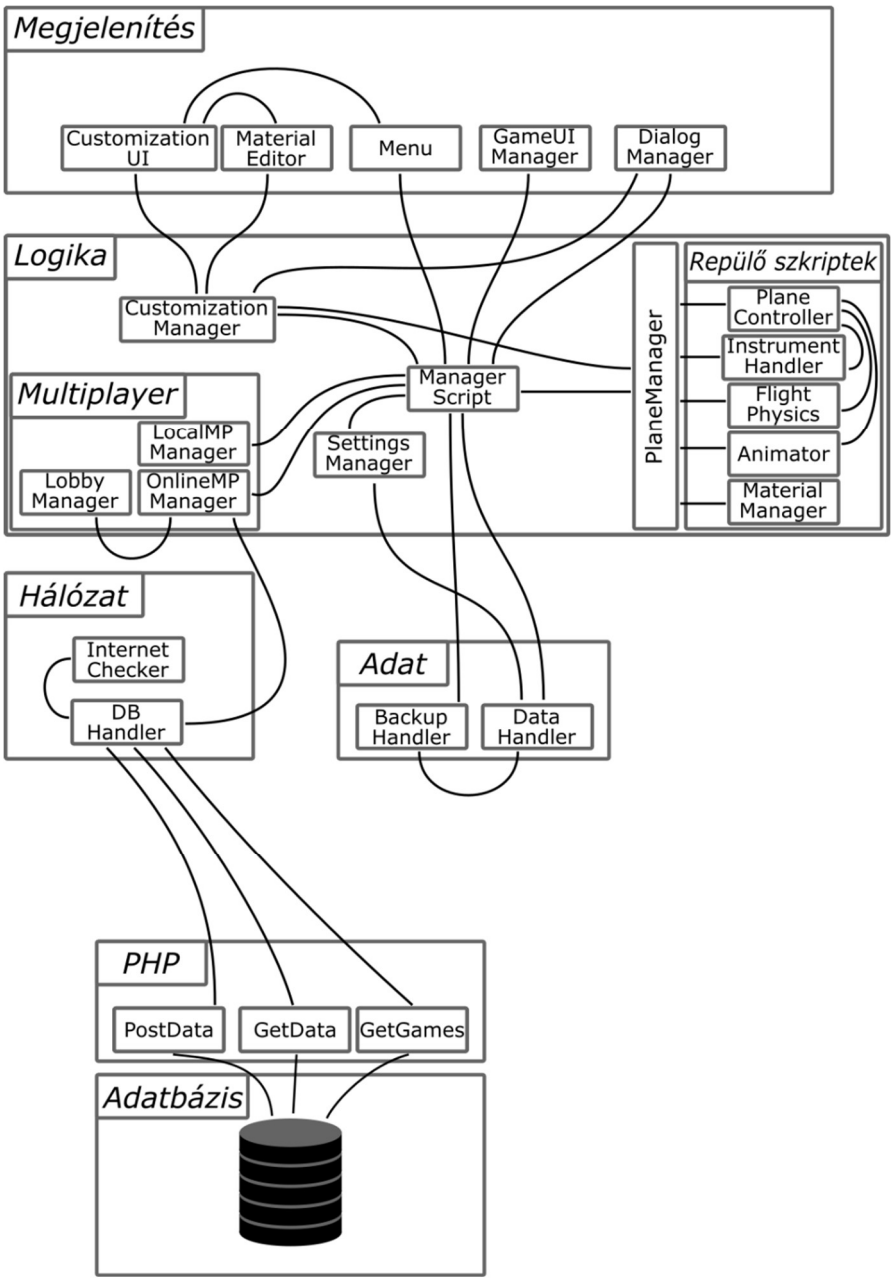
A grafika sok megoldandó problémát felvet. A terep megjelenését, amin a játékosok repülnek, erdőkkel teszem szebbé. Fák elhelyezésére a Unity-ben beépített eszköz van. Ebbe importáljuk a fa modelljét, és tömegesen el tudjuk helyezni a terepen. Ennek

segítségével dús erdőket lehet létrehozni. Nagy előnye, hogy a távoli fákat kétdimenziós lapokkal helyettesíti, amelyekre kirajzolja a fa modelljét, ezzel optimalizálva a renderelést. Azonban bizonyos fa mennyiség fölött már ez sem elég. A repülőgép szimulátor játéktere hatalmas, így óriási területet kell dús erdőkkel beborítani. Az az út sem járható, hogy csak a közeli fákat rajzolja ki a játék, mert a játékos magasan a felszín felett fog repülni, így a terep legtávolabbi pontjait is belátja, ezért az ott található fákat is meg kell jeleníteni. Mindez odavezet, hogy rendkívül sok fa kirajzolása szükséges. A beépített eszköz képes ennyi fa megjelenítésére, azonban a helyettesítő lapokat mindig be kell forgatni a kamera irányába, hogy helyesen megjelenjenek. Ez a procedúra viszont ilyen nagy erdőméret mellett jelentősen lelassítja a játékot, olyannyira, hogy az diavetítéssé redukálódik.

A fák beforgatása tehát szűk keresztmetszet. Ennek orvoslására ezt a feladatot a GPU-ra hárítom, amin hatékonyan lehet párhuzamosítani. Ez azért lehetséges, mert a vertex shaderek manipulálni tudják a modellek csúcsait, így a fákat a shaderek is be tudják forgatni a játékos felé, ezzel levéve a terhet a CPU-ról.

5. TERVEZÉS

A projekt két fő részből áll. A kliensből, ami a játékos eszközén futó játék és a szerverből, ami egy PHP végpontokon keresztül elérhető adatbázis. A vázlatos részeket az 5.1. ábrán lehet nyomon követni.



5.1. ábra: Vázlatos terv

5.1. Kliens

A kliens a felelősségi körök alapján további négy részre bontható. Ezek a játéklogika, megjelenítés, perzisztálás, és hálózati kommunikáció.

5.1.1. Logika

A logika központi része a *ManagerScript*. Ez indítja el és állítja le a játékot, illetve ez továbbítja a kéréseket az egyes funkciókért felelős szkripteknek (például biztonsági mentés készítés).

A repülőgépek irányításáért több szkript felelős. Erre példa a *PlaneController*, ami a játékostól kapott bemenetet dolgozza fel (vízszintes, függőleges kormányzás, csűrés stb.). Ezeket továbbítja az *Animator*-nak, ami bemozgatja megfelelő kormányfelületeket. Az *InstrumentHandler* felelős a műszerek működtetéséért. A repülés logikája a *FlightPhysics* szkriptben fut. Ez a repülőgép pozícióját, sebességét, orientációját, és felhasználói bemenetet ismerve kiszámolja a repülőgépre ható erőt és alkalmazza a merevtestre.

Ezeket a repülőgép szkripteket a *PlaneManager*-en keresztül lehet elérni. Ez tartja számon, hogy éppen melyik repülőgép az aktív, így a hívó félnek ezzel nem kell törődnie.

A gépek személyre szabását a *CustomizationManager* végzi. Ez tartja számon a felhasználó által definiált matériákat, és rendeli hozzá az egyes repülőgépekhez. A matéria tényleges alkalmazása a *MaterialManager* feladata, amit a *PlaneManager*-en keresztül lehet elérni.

A többjátékos mód levezénylése ki van szervezve az *OnlineMPmanager* és *LocalMPmanager* szkriptekbe.

5.1.2. Megjelenítés

A menüben történő navigálást a *Menu* szkript kezeli. Ez váltogat az egyes felületek között (főmenü, beállítások, repülőgép választás, biztonsági mentés stb.). A repülőgépek személyre szabásánál használt felületért a *CustomizationUI* a felelős, ami kommunikál a *CustomizationManager*-rel.

Az elindított játék felhasználói felületét a *GameUIManager* irányítja. A kiválasztott irányítástól és az éppen aktív repülőgép képességeitől függően jeleníti meg vagy rejti el a UI elemeket (például a fékszárnyat állító kar csak fékszárnnyal rendelkező gépek esetén jelenik meg). E mellett kezeli a kistérképet és a játék szüneteltetésénél megjelenő felületet.

5.1.3. Perzisztálás

A fájlkezelés a *DataHandler* feladata. A beállításoktól kezdve a játékos egyenlegén át az egyedi matériákig, minden adatot ez ment le és tölt be. Ismeri a célfájlokat, és hogy milyen adatot melyikbe kell írni.

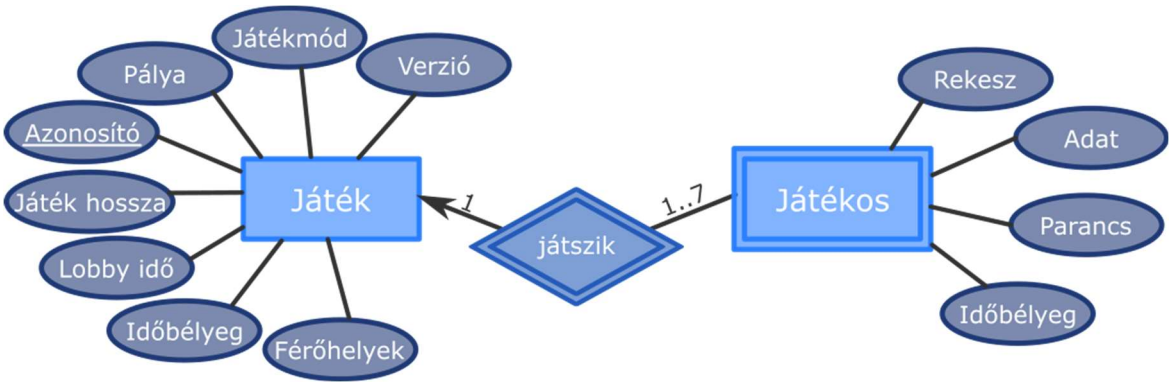
Logikailag a biztonsági mentés készítéséért felelős *BackupHandler* is ide tartozik, azonban a fájlíráshoz ez is a *DataHandler*-t hívja meg.

5.2. Szerver

A szerver egy adatbázisból, és az ez elé felállított PHP biztonsági kapuból áll.

5.2.1. Adatbázis

Az adatbázis szerkezetének tervét az 5.2-es ábrán lehet megtekinteni.



5.2. ábra: Az adatbázis EK diagramja

A legfontosabb egyed a *Játék*, ami egy játékszobát reprezentál, amire a játékosok csatlakozhatnak. A tulajdonságait az 5.1-es táblázat ismerteti:

Tulajdonság	Magyarázat
Azonosító	A játéktábla elsődleges kulcsa
Verzió	A játékosok adatainak formátumára vonatkozó verziószám
Játékmód	Milyen játékot játszanak a játékosok (szabad repülés / csillag gyűjtés)
Pálya	A játék helyszíne
Férőhelyek	Hány játékos csatlakozhat maximum
Játék hossza	A játék hossza másodpercben
Lobby idő	A lobbyban töltött idő hossza
Időbélyeg	A játék kezdetének időpontja

5.1. táblázat: Játék egyed tulajdonságai

A *Játékos*ok tulajdonságai az 5.2-es táblázatban találhatóak.

Tulajdonság	Magyarázat
Rekesz	Megadja, hogy a játék melyik rekeszében játszik.
Adat	A repülőgép kirajzolásához szükséges adatok (pozíció, orientáció, matériák stb.)
Parancs	Utasítás, amit a többi kliensnek végre kell hajtania
Időbélyeg	Az utolsó adat megosztás időpontja

5.2. táblázat: Játékosok tulajdonságai

A *Játékos* egy gyenge egyed, aminek az azonosításához a szoba azonosítójára, és a játékos által lefoglalt rekeszre van szükség.

5.2.2. PHP kapu

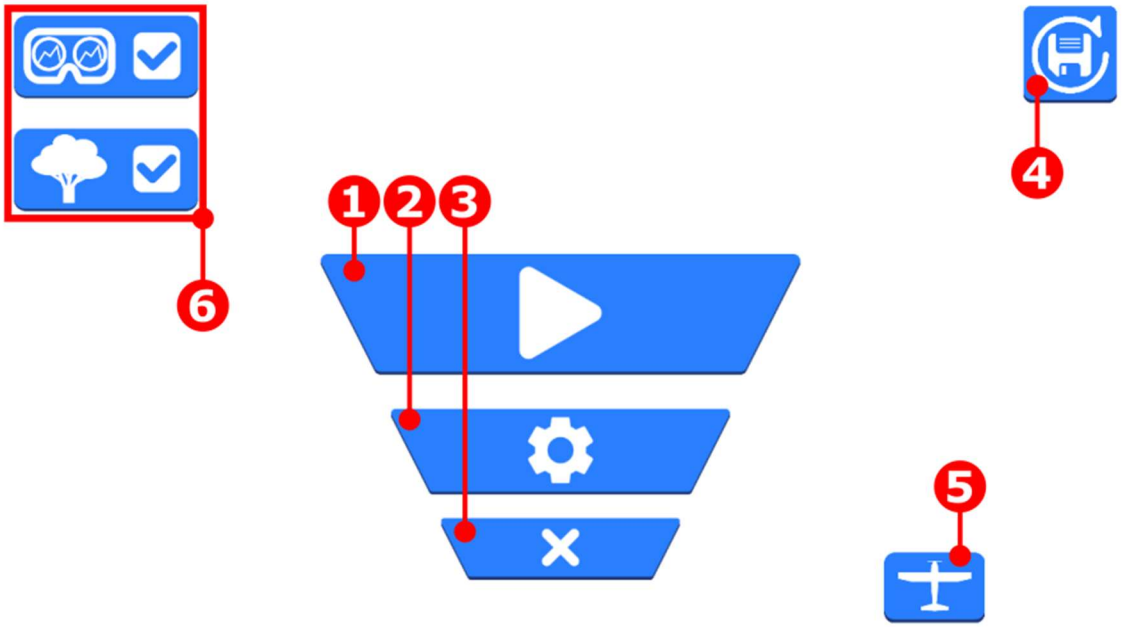
A PHP kapu elsődleges feladata az adatbázis védelme. Ehhez két fontos lépést tesz. Szűri, hogy kik férhetnek hozzá, és meghatározza, hogy ők mire és hogyan használhatják az adatbázist.

Ezen végpontok, amennyiben ártalmas forgalomra utaló nyomokat találnak, mint például nem egyező alkalmazás aláírás vagy tiltott karakterek a paraméterekben (potenciális SQL injekció), az eszközt tiltólistára teszik, és megtagadják az adatbázis elérést. Ezen felül figyelik, hogy támogatott játékverzióról érkezett-e a kérés (arra az esetre, ha egy verzióban olyan hibákra derül fény, amik zavart okozhatnak a multiplayer működésében, és emiatt tiltásra van szükség).

Ha a felhasználó jogosult az adatbázis elérésére, akkor a kapott paraméterekből összeállítja az SQL kérést, amit lefuttat, és a választ megjeleníti. Mivel az SQL utasítás a szerveren van összeállítva, méghozzá validált paraméterekkel, kikényszeríthető az adatbázis rendeltetésszerű használata.

6. FELHASZNÁLÓI FELÜLET TERVEK

Főképernyő



6.1. ábra: A főképernyő terve

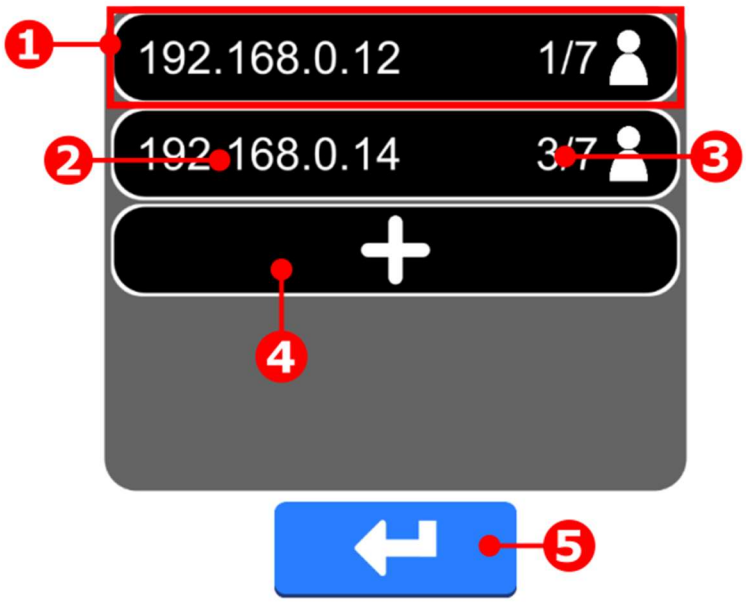
A főmenü részeit az 6.1. ábrán lehet végig követni.

A játék gombbal (1) indítható el a játék (még a játékmódot is ki kell majd választani). Ez a legfontosabb gomb. A játékos ezt fogja a leggyakrabban megnyomni. Emiatt ez is a legnagyobb. A beállítások gombbal (2) különféle beállítások érhetőek el. Ilyenek a hangerő és az irányítás. A teljesség érdekében egy kilépés gomb (3) is elhelyezésre került.

A mentés és visszaállítás gombbal (4) érhető el a biztonsági mentés menüje. Itt lehet a játék adatát kimenteni egy fájlba, amit később, akár egy másik eszközön is vissza lehet állítani.

A hangár gombon keresztül (5) lehet váltogatni a repülőgépek között. Az átfestés felület is innen érhető el. A bal felső sarokban található néhány kapcsoló (6), amivel a játék funkcióit lehet ki-be kapcsolgatni. Ilyen például a VR mód, vagy a fák megjelenítése. A fák sokat dobnak a megjelenésen, azonban néhány eszközön lelassulhat miattuk a játék, így szükséges lehet azokat kikapcsolni.

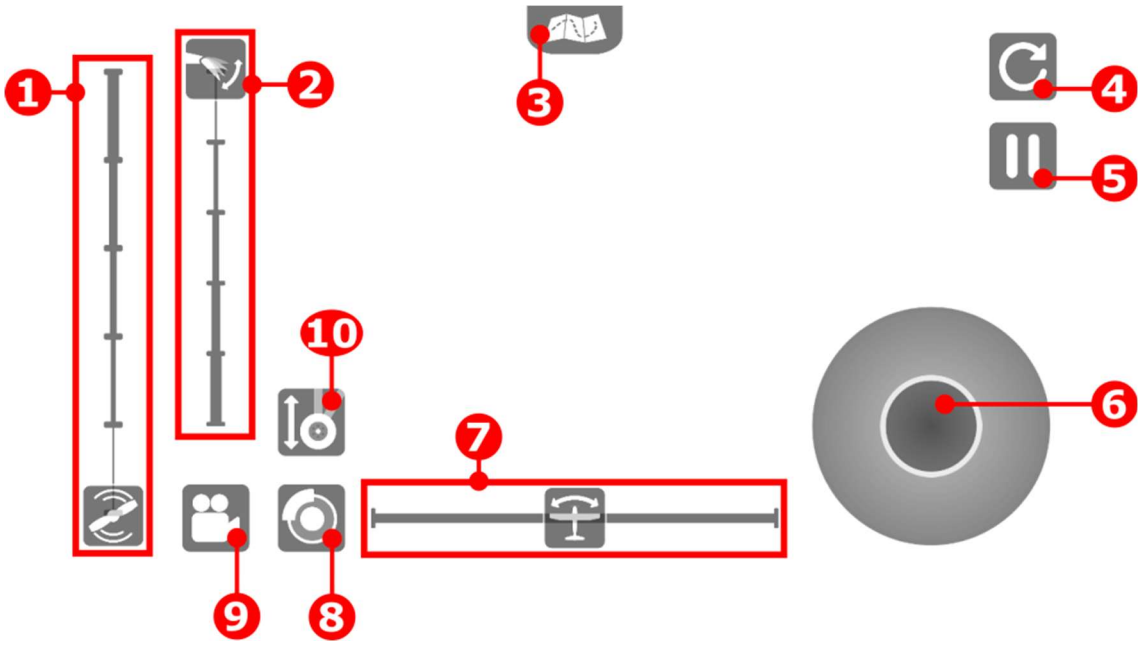
Lokális multiplayer lista



6.2. ábra: A lokálisan futó játékok listája

Az 6.2. ábrán látható felület egy listát tartalmaz az elérhető játékszobákról. Amennyiben már hosztolnak játékot a lokális hálózaton, akkor az egy listaelemként fog megjelenni (1). Ez tartalmazza a hoszt IP címét (2), és a csatlakozott játékosok számát (3). Új játék hosztolásához az utolsó listaelemet (4) kell kiválasztani.

Játék UI



6.3. ábra: Alapértelmezett irányítási felület

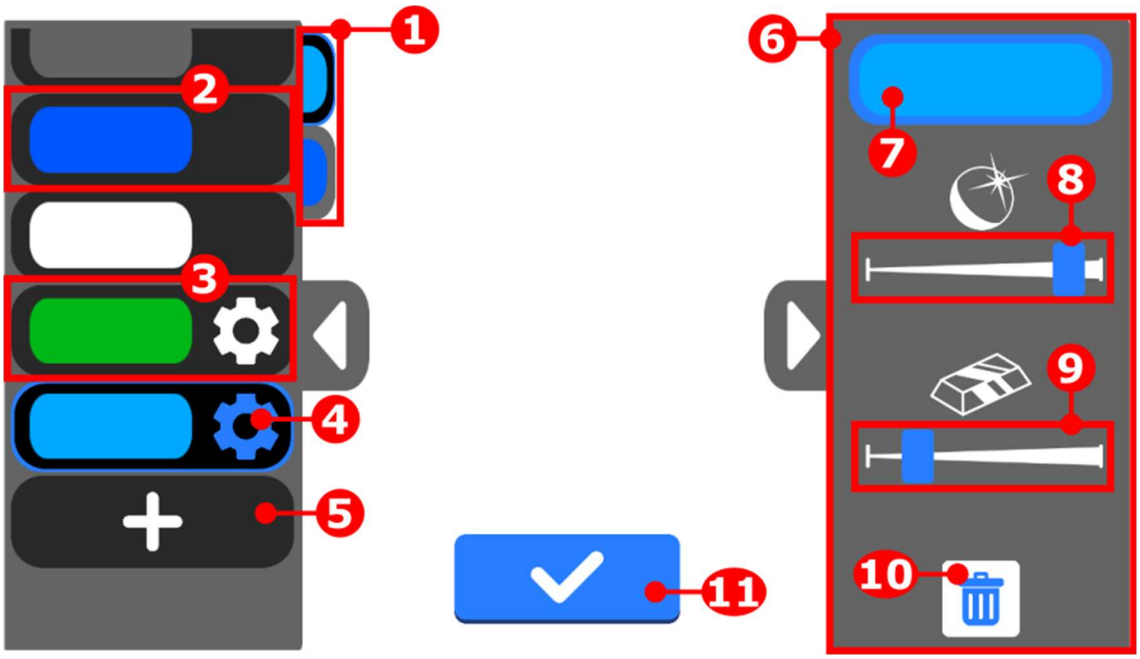
Az 6.3. ábrán az alapértelmezett irányítási felület látható.

Az irányításban a legfontosabb elemek a gázkar (1) és a botkormány (6). A gázkar függőleges mozgatásával szabályozhatóak a hajtóművek. A botkormány a csűrőket és a magassági kormányt állítja. Csupán ezek használatával is már irányíthatóvá válik a repülő. A fordulások javításához használatos az oldalkormány (7), hogy a gép ne csússzon se befelé se kifelé. A fékszárnyakkal rendelkező repülőkön, az azokat állító kar (2) is megjelenik.

A gombok közül a leghasználatosabb a nézetváltás gomb (9). Ezzel lehet pilótafülke és kívülnézet között váltani. A fék gomb (8) a fékek mellett, (ha a gép rendelkezik velük) a féklapokat is irányítja. Amennyiben a futóművek behúzhatóak, azokat a 10-essel jelölt gombbal lehet kontrollálni. Az újrakezdés gomb (4) visszarakja a játékost a kiinduló pozícióba. Ez elsősorban akkor használatos, ha a repülőgép összetört. A szüneteltetés gomb (5) hozzáférést ad a megjelenést szabályozó kapcsolókhoz, a látótávolság szabályzójához és a hangerő beállításokhoz. Egyjátékos módban szünetelteti a játékot.

A képernyő tetején található a térkép fül (3), ami egy kistérképet nyit meg. Ez elsősorban többjátékos módban hasznos, hogy a játékosok könnyebben megtalálják egymást, de offline játék esetén is elérhető.

Átfestő



6.4. ábra: Repülő átfestő menü

A repülőgépek kinézetét két matéria határozza meg. Ezeknek megváltoztatható a színük, sőt még azt is meg lehet határozni, hogy mennyire legyen fémes hatású és hogy fényes vagy matt legyen. A felhasználói felület az 6.4. ábrán látható.

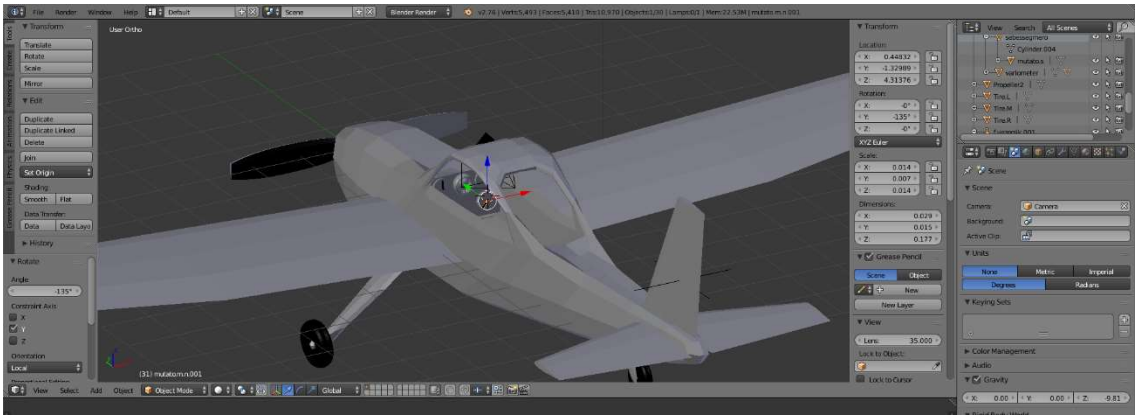
A matéria fülek (1) segítségével választhatja ki a játékos, hogy melyik anyagfoglalatot szeretné módosítani. Baloldalt jelennek meg a definiált matériák. Egy ilyenre rányomva, az anyag hozzárendelésre kerül a kiválasztott foglalathoz. A lista tetején (2) a repülőkön alapértelmezetten használt matériák jelennek meg. Ezeket nem lehet módosítani. Alattuk találhatóak az egyedi (felhasználó által létrehozott) anyagok (3). Ezeket a szerkesztés gombbal (4) lehet módosítani. Új anyagdefiníciók létrehozásához a plusz gombra (5) kell nyomni.

A szerkesztés gomb hívja elő a szerkesztői felületet (6). Itt lehet módosítani az anyagszínt (7), azt, hogy mennyire legyen fényes, avagy matt (8), és hogy mennyire keltsen fémes hatást (9). Ha az anyagdefiníciót törölni szeretnénk, azt a kuka gombbal (10) tehetjük meg. Ha végeztünk az átfestéssel, a rendben gombra (11) nyomva térhetünk vissza az előző nézethez.

7. MEGVALÓSÍTÁS

7.1. Modellezés

A projektet a kezdő repülőgép megalkotásával kezdtem. A modellt az ingyenes *Blender* 3D tervező programban készítettem, amint az a 7.1-es ábrán is látható. A kormányfelületek és a kormány, továbbá a műszerek külön-külön kialakításra kerültek, annak érdekében, hogy a játékban meg lehessen ezeket animálni.



7.1. ábra: A kezdő repülőgép modellezése Blenderben

Ugyancsak *Blender*ben készült a terep is, amin a játékosok játszanak. Egy szigetet formáztam meg, mert így a modell széleit el lehet rejteni a víz alá, lásd 7.2. ábra.

7.2. ábra: A terep modellezése Blenderben

A modellek készítésénél fontos szempont volt, hogy a sokszögek számát alacsonyan tartsam, az alacsony teljesítményű mobileszközök érdekében. A 7.1-es ábrán jól láthatóak

a repülőt alkotó téglalapok. A téglalapok határait a játékban simított megvilágítással rejtem el.

7.2. A repülés fizikája

A fizikának két fő része van. A felhajtóerő és a légellenállás.

Felhajtóerő

A felhajtó erő kiszámításához Fábián András könyvéből tanult képletet használom (1). [9]

(1) $$Y = C_y \cdot \frac{\varrho \cdot v^2}{2} \cdot A$$

Ahol: Y = felhajtóerő

Cy = felhajtóerő-tényező

ϱ = a levegő sűrűsége

v = a légáramlás sebessége

A = szárnyfelület

Ebben a képletben a legnagyobb problémát a Cy kiszámítása okozza. Ennek az értékét mérésekkel lehet meghatározni, de nekünk egy képlet kell.

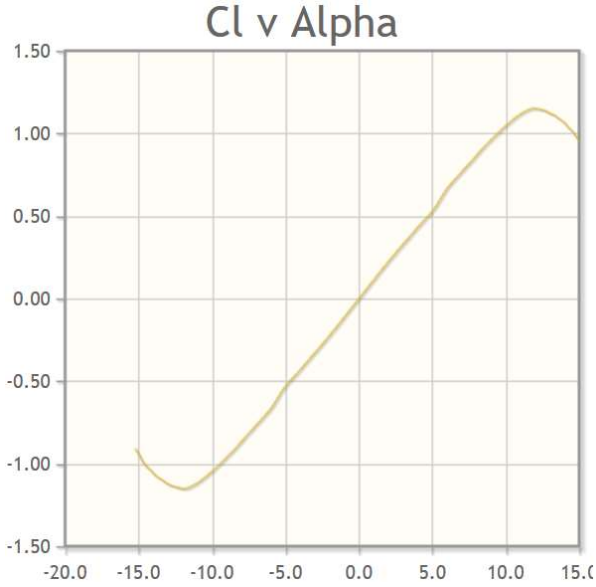
A legkézzelfoghatóbb képletet (2) Brian J. Cantwell Alkalmazott Aerodinamika jegyzetében találtam [10]. Igaz, elliptikus szárnyakra.

(2) $$\frac{C_y}{2\pi\alpha} = \left(\frac{A_R}{2+A_R}\right)$$

Ebből az egyenletből ki lehet fejezni a felhajtóerő-tényezőt (3).

(3) $$C_y = \left(\frac{A_R}{2+A_R}\right) \cdot 2\pi\alpha$$

Ez az egyenlet egyenes arányosságot definiál a felhajtóerő-tényező és az állásszög között. Ahogyan azonban az 7.3. ábrán láthatjuk, ez csak alacsony állásszögekre igaz.

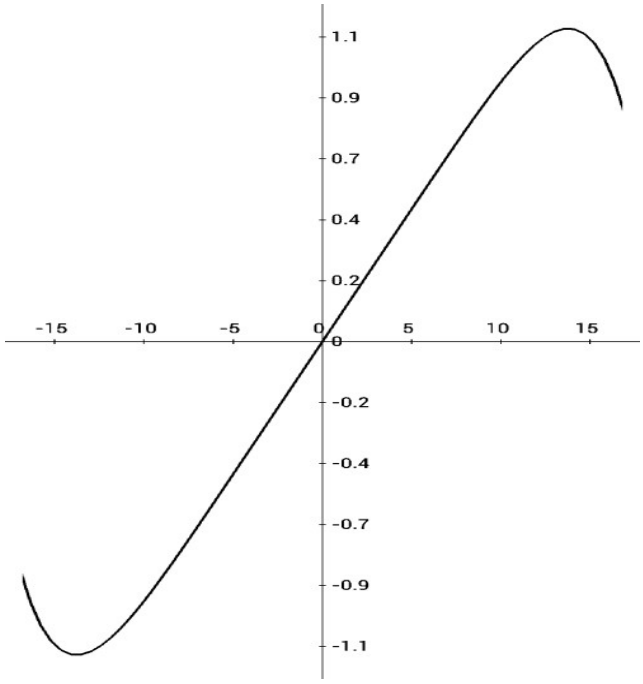


7.3. ábra: A felhajtóerő-tényező az állásszög függvényében. Forrás: http://airfoiltools.com/airfoil/details?airfoil=hq010-il#polars

Korrigálásképpen az egyenletet megszoroztam egy magasfokú polinommal (4), ami a [-10;10] tartományon közel 1-et vesz föl, majd hirtelen elkezd csökkenni, lehúzva ezzel a kezdeti egyenest.

(4) $$-(0.05x)^6 + 1$$

Az eredmény a 7.4. ábrán látható.



7.4. ábra: Korrigált felhajtóerő-tényező függvény

A projektben ezt a korrigált formulát alkalmazom.

Légellenállás

A légellenállás képlete (5) nagyban megegyezik a felhajtóerő képletével. Itt azonban *Cy* helyett *Cx*, azaz légellenállás-tényező szerepel. [9]

(5) $$X = C_x \cdot \frac{\varrho \cdot v^2}{2} \cdot A$$

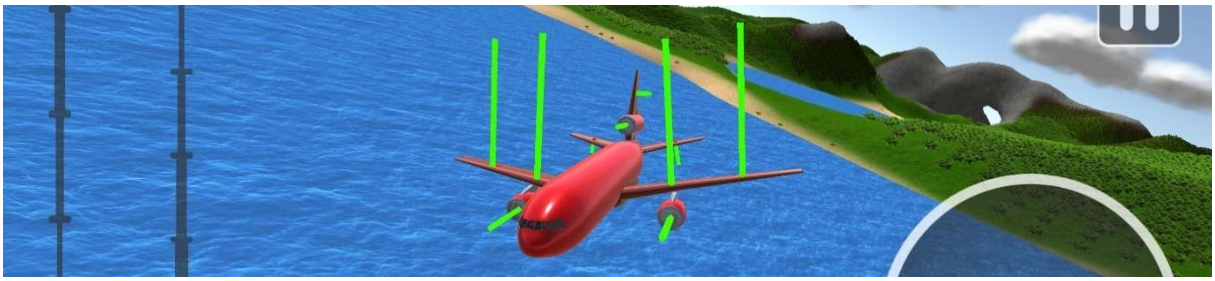
A szárnyokon keletkező, indukált légellenállási-tényező (*Cxi*) kiszámítására a következő képlet alkalmazható (6). [10]

(6) $$C_{xi} = \frac{C_y^2}{\pi A_R}$$

Ezen formulákkal modellezem a repülés fizikáját.

A propellerek által adott tolóerő kiszámítására is alkalmazható a fenti képlet, ebben az esetben azonban a *v* a propeller forgási sebességéből adódik.

A felhajtóerő és a légellenállás a szárnyakon 4, a vízszintes vezérsíkokon 2, a függőleges vezérsíkon 1 ponton lesz kiszámítva, amint a zöld csíkok mutatják a 7.6. ábrán.



7.5. ábra: A repülőre ható erők

A légellenállás ezen pontokon kívül a géptörzsre és a behúzható kerekekre is ki van számítva.

7.3. Egyjátékos mód

Az alkalmazás alapvető funkciója az egyjátékos, azaz offline mód. A játékos a 7.2-es ábrán látható szigeten repülhet szabadon. Ez tökéletes lehetőséget nyújt arra, hogy ügyesedjen a repülőgépek irányításában. A terepen több reptér is található, amelyeken gyakorolni lehet a leszállást.

7.4. Többjátékos mód

Az online multiplayer működési elve a következő:

Minden játékos feltölti az adatait (pozíció, sebesség, orientáció, stb.) az adatbázisba. Egy kis idő múlva pedig letölti a többiekét, majd szintén kis idő múlva elölről kezdi. Körülbelül 2 fel- és 2 letöltés történik másodpercenként (játékosonként).

Annak érdekében, hogy az is eldönthető legyen, ki aktív és ki nem, az adatbázis az utolsó feltöltés időbélyegét is tárolja.

Minden kliens a kapott adatok alapján az aktív játékosok gépeit ki tudja rajzolni a kapott koordinátákra a kapott orientációval. Az adatok a repülők sebességét is tartalmazzák, ami alapján azok tovább animálhatóak a letöltések között, hogy a mozgás kevésbé legyen szakadozott.

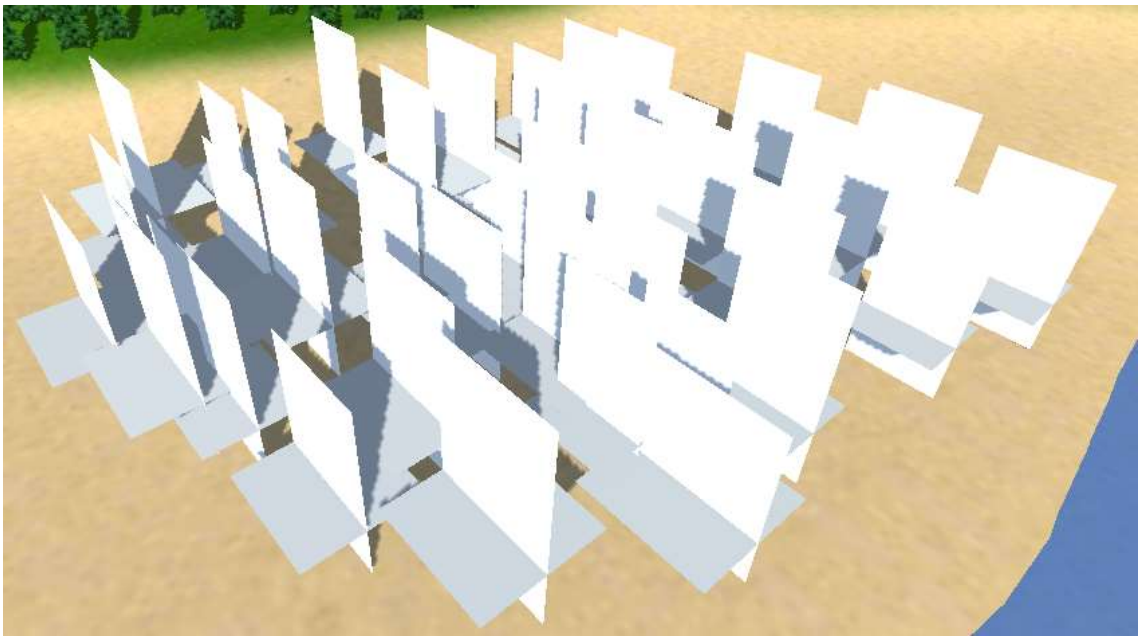
Ez a módszer (ilyen formán) csak akkor használható, ha a játékosok kizárólag a saját gépeik állapotát változtathatják. Azonban a környezetben okozott változás így nem jelenik meg a többi kliensen. Ezt használom ki a Free Flight játékmódnál, ahol a játékosok egymástól függetlenül gyűjtögethetik a lebegő érméket.

A Star Collector-nál azonban, ha egy játékos felszedett egy csillagot, azt a többi játékos játékteréből is el kell távolítani. Erre a célra a játékos adata és időbélyege mellett még egy parancsmező is bevezetésre került. Ha a játékos begyűjt egy csillagot, akkor azt parancsban jelzi a többieknek. Például: „Collect;32,12” ahol a „Collect” a parancskód a 32 és a 12 pedig az attribútumok (itt a csillagok sorszámai). A kliensek ezt a parancsot hallva a 32-es és a 12-es csillagokat deaktiválják.

7.5. Grafika optimalizálása

Erdők

Az erdőkhöz először összeállítottam 3D-s modelleket. Ezek a modellek téglalapokból épülnek föl. Minden fa egy vízszintes és egy függőleges téglalapból áll, amit a 7.7-es ábra szemléltet. A függőleges lapok mindegyike a Z tengely negatív irányába néz alapértelmezetten, de amikor kirajzolásra kerül a sor, a vertex shader beforgatja ezeket, hogy a játékosra nézzenek. A fragment shader pedig a már jó irányba álló lapokra rajzolja ki a fák oldalnézetét. A forgatás azonban csak a függőleges tengely körül történik, hogy ne dőljenek a fák. Emiatt szükséges egy vízszintes lap is, amire a fa felülnézete lesz kirajzolva. A végeredmény a 7.8-as ábrán látható.



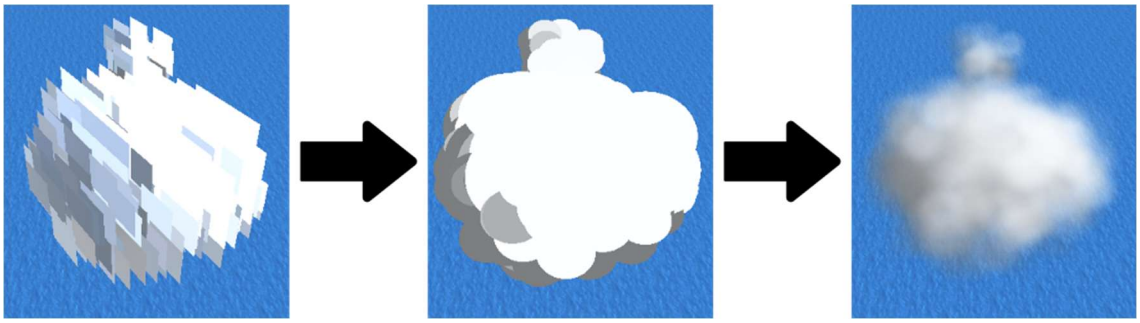
7.6. ábra: Az erdő nyers modellje

7.7. ábra: Shader varázslat nyers modellen

Felhők

Volumetrikus gomolyfelhőket az erdőkhöz hasonlóan alkotok. A koncepció ugyanaz. Legenerálok egy 3D-s modellt, ami lapokból áll (7.9. ábra bal oldala). Ezeket a lapokat egy shader beforgatja a játékos irányába, és mindegyikre kirajzol egy kört (7.9. ábra közepe). A körök a peremük felé egyre áttetszőbbek, hogy összemosódjanak (7.9. ábra jobb oldala). E mellett a lapokra ki van számítva egy vektor, ami megközelítően megadja

a felhő felszínének normálvektorát azon a ponton, ahol az adott lap elhelyezkedik. Ez a vektor alapján a shader árnyékolni tudja a felhőt, hogy a napsütötte rész világosabb legyen.



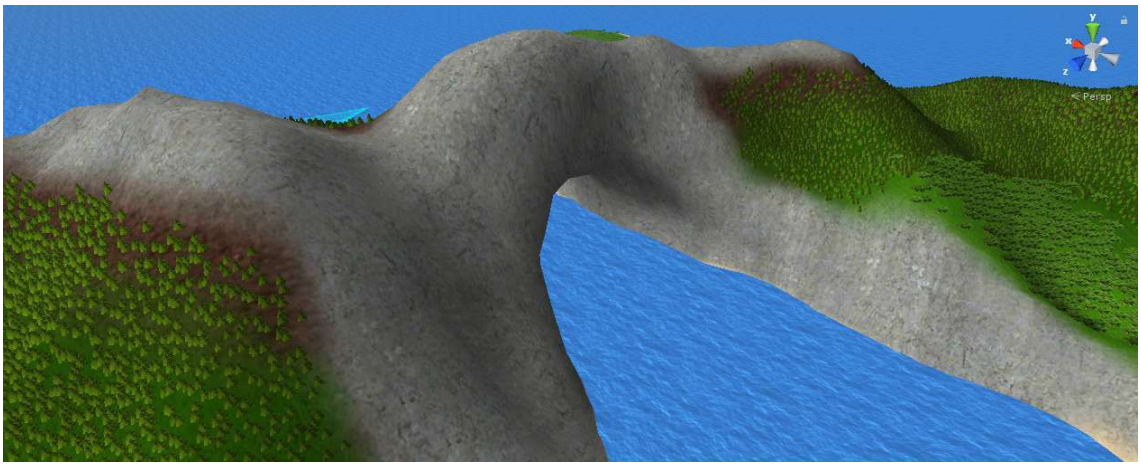
7.8. ábra: Nyers modell felhővé alakulása

Az eredmény valósághű volumetrikus felhők (7.10. ábra).

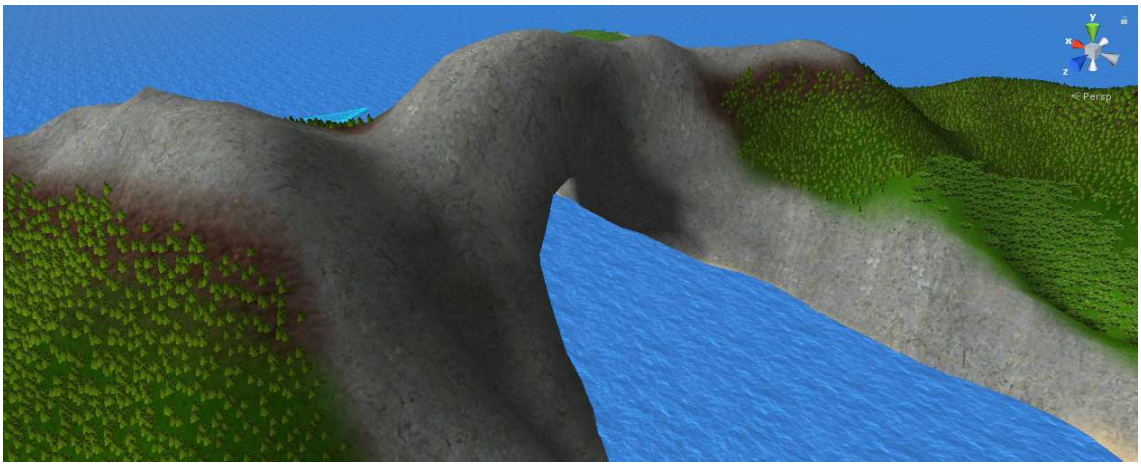
7.9. ábra: Végleges felhők

Világítás

A szigeten, ahol a játék játszódik, kialakítottam egy sziklaívet, amin át lehet repülni. Ha az ív nem vet árnyékot a sziklafalra, szürreálisnak fog tűnni, ahogyan a 7.11-es ábrán is látható. Ahhoz, hogy valósághűnek látszódjon, az ívnek árnyékot kell vetnie a sziklafalra. A valós idejű árnyékok azonban erőforrásigényesek. A játékmotor által előállított fénytérkép pedig tárhelyigényes, és lelassítja a szerkesztőfelületet. Ennek érdekében egy vertex alapú fénytérképet számítok ki, amivel meghatározom, hogy mely vertexek vannak kitakarva és melyek nem. Mivel a terep modelljében alacsony a sokszögek száma, ez a fénytérkép kis méretű, és gyorsan elkészül. Az eredményt a 7.12-es ábra mutatja.



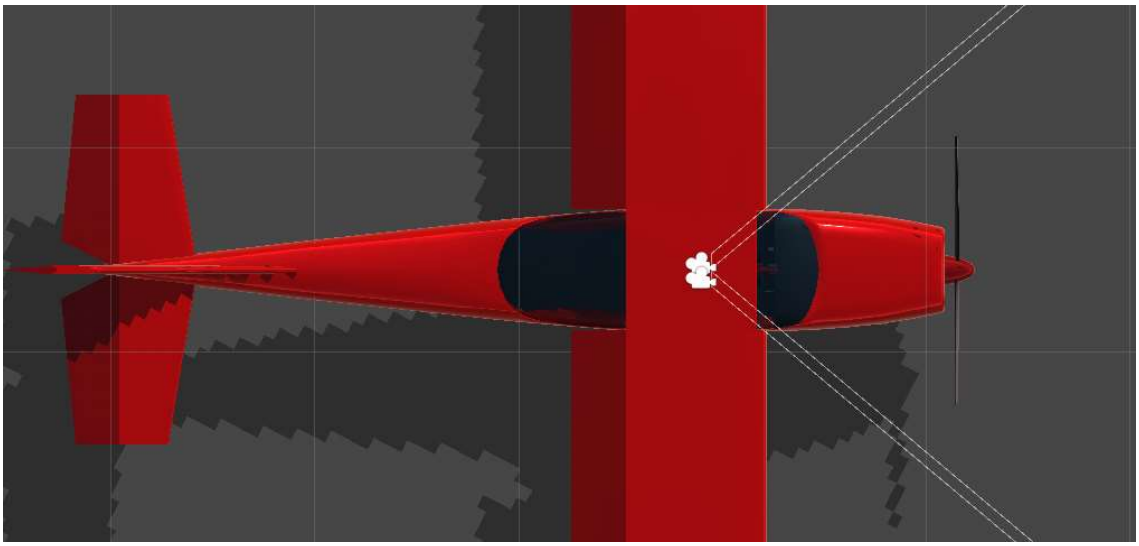
7.10. ábra: Sziklaív fénytérkép nélkül



7.11. ábra: Sziklaív gyors fénytérképpel

7.6. Virtuális valóság

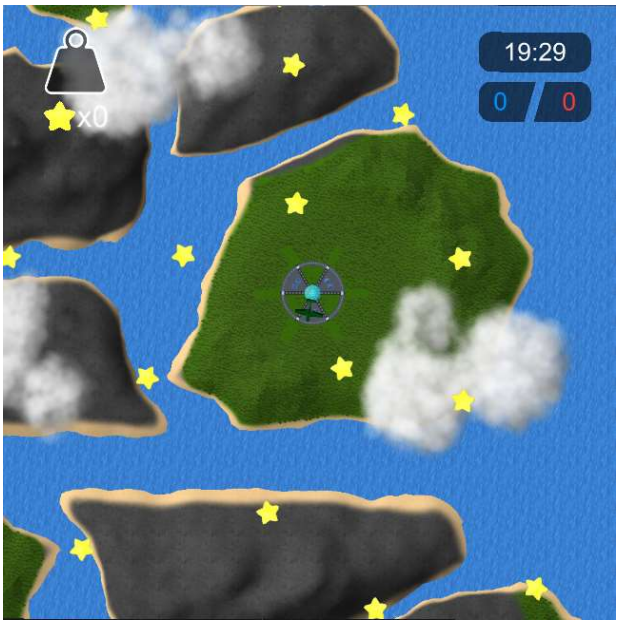
VR módban két egymás melletti kamera is elkészíti az adott képkockát. Az egyik a jobb, a másik pedig a bal szem szemszögéből, majd a képernyőt ketté felezve a két kép egymás mellett jelenik meg. A kamerák elhelyezkedését szemlélteti a 7.13. ábra.



7.12. ábra: A kamerák elhelyezkedése VR módban

Itt a nehézséget a UI elemek elhelyezése okozta. VR módban a játékos kontroller segítségével irányítja a repülőgépét, így az irányításhoz használt csúszkákra és gombokra nincsen szükség, a kistérképre viszont van, továbbá kompetitív játékmód esetén a csapat pontszámát és a hátralévő időt is meg kell jeleníteni. Ennek érdekében a pilótaülés fölött került elhelyezésre egy panel, amin megjelenik a térkép (lásd 7.14. ábra). Továbbá a játékmódhoz szükséges UI elemek is ezen jelennek meg (lásd 7.15. ábra).

7.13. ábra: A térkép elhelyezkedése VR módban

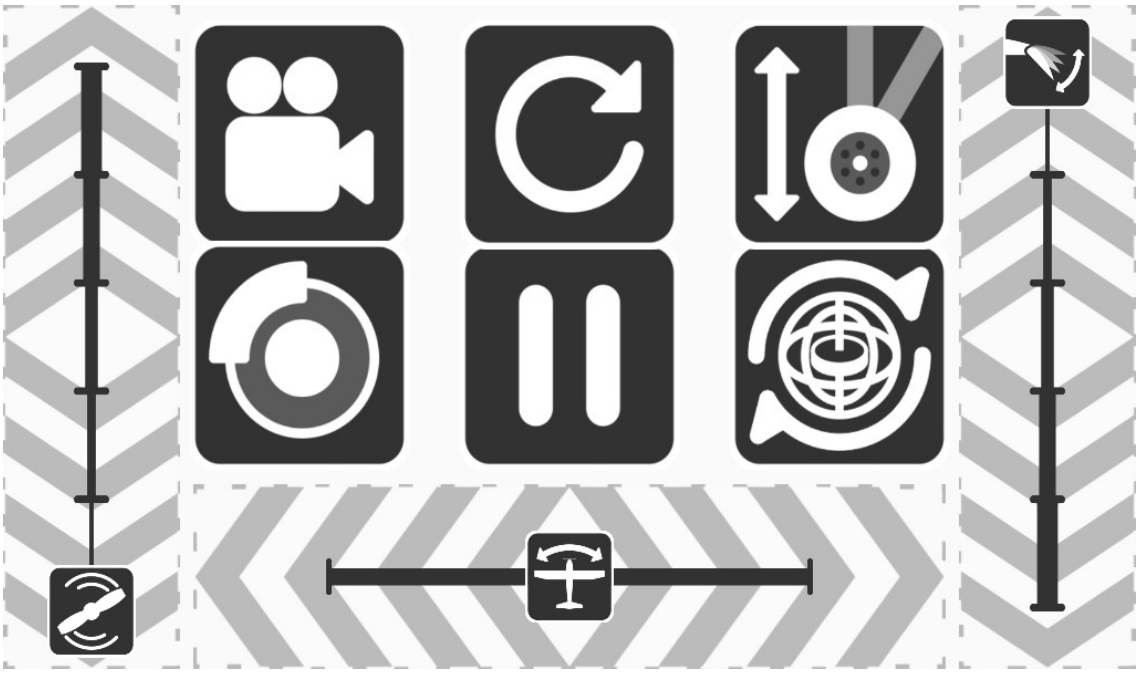


7.14. ábra: UI elemek a térképen

Így a térkép megtekintéséhez a játékosnak csak egyszerűen le kell hajtania a fejét.

7.7. Távirányító alkalmazás

Elsősorban a virtuális valóság mód kiegészítéseként készítettem egy távirányító alkalmazást az Android Studióban. A felhasználói felületet úgy alakítottam ki (lásd 7.16. ábra), hogy a gombok és a csúszkák akkor is könnyen eltalálhatóak legyenek, ha a játékos nem tud lenézni a képernyőre. Az elemek elhelyezkedését természetesen memorizálni kell.



7.15. ábra: Távirányító alkalmazás felülete

A távirányító egy TCP socketen keresztül kommunikál a szimulátorral, periodikusan értesítve azt, hogy mely gombok vannak lenyomva, és hogy a csúszkáknak mi az állapota.

Azért döntöttem ezen megoldás mellett, mert mind a bluetooth mind pedig a WiFi direct kapcsolat kiépítéséhez az alkalmazásnak engedélyt kell kérnie a helymeghatározáshoz, ami bizalmi problémákat vet fel.

7.8. Szerver

Adatbázis

A tényleges táblák megalkotásánál a legfontosabb szempont a lekérések gyorsítása. Emiatt kerülni kell a *Join* műveletet. Ez azt jelenti, hogy minden szükséges adatnak ugyanazon táblában kell megtalálhatónak lennie. Ennek érdekében az adatbázist denormalizáltam. A szobák mezőit kibővítettem játékosok mezőivel. Így nem szükséges join műveletet végrehajtani a játékszobák és a játékosok között. Az eredményt az 7.17-es ábra mutatja.

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra
1	GAMEID	int(6)			Nem	*Nincs*		AUTO_INCREMENT
2	VERSION	int(11)			Nem	*Nincs*		
3	MAP	int(1)			Nem	1		
4	GAMEMODE	int(1)			Nem	1		
5	TIME	timestamp			Igen	current_timestamp()		
6	SLOTS	int(1)			Nem	4		
7	DEVONLY	tinyint(1)			Nem	0		
8	LOBBYTIME	int(4)			Nem	0		
9	GAMELENGTH	int(4)			Nem	600		
10	STARTTIME	timestamp			Nem	current_timestamp()		
11	PLANE0DATA	varchar(140)	latin1_swedish_ci		Igen	x		
12	PLANE0CMD	varchar(100)	latin1_swedish_ci		Nem	x		
13	PLANE0TIME	timestamp			Igen	current_timestamp()		
14	PLANE1DATA	varchar(140)	latin1_swedish_ci		Igen	x		
15	PLANE1CMD	varchar(100)	latin1_swedish_ci		Nem	x		

7.16. ábra: A Játék tábla szerkezete

PHP kapu

Az adatbázis elé felállítottam egy PHP kaput, ami elrejti az adatbázist a kliensektől, így védve azt.

A http kérések gyorsítása érdekében kerülöm az átirányításokat. Emiatt nem mindig egy végpont kerül meghívásra, ami a kéréstől függően átirányít egy másikhoz, hanem funkciókként külön-külön végpontokhoz intéz kéréseket a kliens. A végpontokat, és azok funkcióját a 7.1-es táblázat tartalmazza.

Végpont	Funkció
getgames.php	Az elérhető szobák adatait adja vissza.
postdata.php	A játékos ezzel töltheti fel játékbeli adatait.
getdata.php	Egy adott szobában lévő játékosok dinamikus adatait adja vissza.
getstatic.php	A getdata.php párja. A játékosok néhány adata (repülőgép egyedileg beállított matériái) nem változik a játék alatt, így ezeket elég csak egyszer lekérdezni, ezzel jelentős adatforgalmat megspórolva.
checkrooms.php	Ellenőrzi, hogy van-e szabad szoba. Ha nincs, létrehoz egyet.

poststarttime.php	Lobby-t használó szobáknál, az első csatlakozó játékos hívja meg, ezzel jelezze, hogy mostantól megy a visszaszámlálás a játék kezdéséig.
fixstarttime.php	Amennyiben a lobby betelt, a játék egyből indítható. Ez a végpont a kezdés időpontját korábbra állítja, ezzel megakadályozva, hogy mások csatlakozzanak.

7.1. táblázat: PHP végpontok és funkciójuk

Ezen végpontokon minden kéréskor lefut egy hitelesítés, annak érdekében, hogy a potenciálisan rosszindulatú forgalmat kiszűrjék. A hitelesítés három paraméter alapján történik.

A legfontosabb paraméter az alkalmazás szignatúrája. Amennyiben valaki módosítja a játékot, akkor az aláírása eltérő lesz a hivatalos verzió aláírásától. Ez esetben a végpontok nem fogják kiszolgálni a kérést. E mellett, szintén paraméter az eszköz azonosítója. Ha aláírás eltérés van, akkor az eszköz feketelistára kerül.

A biztonsági mentésekkel történő visszaélés elkerülése érdekében a játék adatainak is van egy azonosítója. Ez egy munkafolyamat azonosító. A játék első indításánál jön létre, és belekerül a biztonsági mentésekbe. A mentések visszaállításánál, ez az azonosító is vissza lesz állítva. Ezzel detektálható az az eset, amikor valaki közzé teszi a biztonsági mentését, és sok játékos ezt felhasználva becstelen előnyre tesz szert, mivel nekik így nem kell megdolgozniuk a repülőgépek megszerzéséért.

7.9. Biztonsági mentések

A biztonsági mentések lehetőséget adnak a játékosoknak arra, hogy az adataikat hordozhassák. Ez azonban annak a veszélyével fenyeget, hogy a mentést átírva tisztességtelen előnyre tesznek szert a többi játékoshoz képest. Emiatt a mentést úgy kell összeállítani, hogy annak integritása ne tudjon észrevétlenül sérülni.

A titkosítás eljárása a következő:

1. Összeillesztem az összes mentendő adatot egy adathalmazzá.
2. Az így kapott adathalmazt titkosítom egy véletlenszerűen generált kulccsal.
3. A titkosított adathalmazban összeszámolom az 1-es biteket.
4. A titkosított adathalmaznak kiszámolom a bitközéppontját (az 1-es bitek helyeinek átlaga).
5. A kulcsot titkosítom, először az 1-es bitek számával.
6. Még egyszer titkosítom a kulcsot a bitközépponttal.
7. Végül a titkosított adathalmazt és a titkosított kulcsot beleírom egy fájlba.

A visszafejtésnek pedig ezek a lépései:

1. Beolvasom a titkosított adathalmazt és a titkosított kulcsot.
2. Kiszámolom a titkosított adathalmaz bitszámát és bitközéppontját.
3. A bitszámmal és a bitközépponttal visszafejtem a kulcsot.
4. A visszafejtett kulccsal visszafejtem az adathalmazt.
5. Végül az adathalmazból kiolvasom az adatokat.

Az integritást az biztosítja, hogy amennyiben a felhasználó módosítja a fájlt, akkor a kiszámolt bitszám és/vagy bitközép is módosulni fog. Ennek következtében a visszafejtés során kapott kulcs helytelen lesz, amivel nem lehet az eredeti adathalmazt helyesen visszaállítani, hanem az értelmetlen lesz. Ezért, ha a visszafejtett adat értelmes, akkor a fájl hiteles.

8. TESZTELÉS

8.1. Funkcionális tesztek

A következő táblázatokban (8.1-, 8.2-, 8.3- és 8.4-es táblázatok) azok a tesztelési forgatókönyvek találhatóak, amelyeken minden verzió kiadásánál végig megyek. A táblázat bal oldalán egy szituáció vagy cselekvés áll, jobb oldalt pedig az elvárt következmény. A táblázatok nem tartalmaznak minden lehetséges forgatókönyvet, de ha ezeken a teszteken megfelel a projekt, akkor magabiztosan állíthatom, hogy a játék alapvetően helyesen működik. Ezeken kívül az új verzióval bemutatott funkciókat is alaposan letesztelem. Példa erre a kistérkép, vagy a repülők átfestése.

Egyjátékos mód tesztjei		
Forgatókönyv	**Elvárt kimenet**	**Siker**
Repülő felgyorsítása, majd felhúzása	a repülő elemelkedik	✓
Repülés közben gáz levétele	siklik	✓
Repülés közben gáz levétel, és az orr felhúzása	átesik	✓
A repülő odacsapása a földhöz	a repülő felrobban	✓
Óvatos leszállás	nem robban fel	✓
Nekifutás egy falnak	felrobban	✓
Belerepülés a tengerbe	lelassul, elmerül, kék köd bekapcsol	✓
Fékszárnyak kinyitása	felhajtóerő megnő	✓
RESET gomb megnyomása	a repülő visszaáll a kiindulási állapotba	✓

8.1. táblázat: Tesztelési forgatókönyv egyjátékos módban

Többjátékos mód tesztjei		
Forgatókönyv	**Elvárt kimenet**	**Siker**
Csatlakozás egy játékhoz	a megfelelő hangárba kerül a repülő	✓
Belerepülés egy pénzérmébe	az egyenleg megnő	✓
Egy másik játékos is csatlakozik	a repülője helyesen megjelenik	✓
Üzenet küldés chat-en	az üzenet megérkezik	✓

Belerepülés egy csillagba	a csillag eltűnik, a repülő súlya megnő	✓
Lezuhanás begyűjtött csillaggal	a csillag visszakerül	✓
Csillaggal sikeres leszállás a bázison	minden csillagért egy pontot kap a csapat	✓
Csillaggal leszállás az ellenséges bázison	nem történik semmi	✓
RESET gomb megnyomása	a repülő visszaáll a kiindulási állapotba	✓

8.2. táblázat: Tesztelési forgatókönyv többjátékos módban

Monetizációs tesztek		
Forgatókönyv	**Elvárt kimenet**	**Siker**
A játék elindítása	betöltődnek reklámok	✓
Reklám elindítása gomb megnyomása	reklám megjelenik	✓
Reklám bezárása	a játékos egyenlege nem változik	✓
Reklám végig nézése	az egyenleg megnő a nyereménnyel	✓
Érme vásárlás gomb megnyomása	a Google Play megerősítő dialógus ablaka megnyílik	✓
Fizetés megszakítása	az egyenleg marad	✓
Fizetés kártyával, ami visszautasítja a tranzakciót	az egyenleg marad	✓
Sikeres fizetés	az egyenleg a vásárolt érmékkel megnő	✓
Kilépés a többjátékos módból 5 perc után	egy reklám jelenik meg	✓

8.3. táblázat: Tesztelési forgatókönyv a hirdetésekre és vásárlásokra

Mentéssel kapcsolatos tesztek		
Forgatókönyv	**Elvárt kimenet**	**Siker**
A játék elindítása	a legutóbb használt repülő az aktív	✓
A játék elindítása	az egyedi matériák és az egyenleg betöltődik	✓
Biztonsági mentés készítése	sikeresen létrejön a mentés	✓
Biztonsági mentés visszaállítása	minden adat visszaáll a mentéskori állapotra	✓
Korábbi mentés visszaállítása	minden adat visszaáll a mentéskori állapotra	✓
Egyenleg megváltoztatása bármi módon	automatikusan készül biztonsági mentés	✓

8.4. táblázat: Tesztelési forgatókönyv a mentésekre

8.2. Unit tesztek

A projekt, jellegéből adódóan nem ideális ahhoz, hogy unit tesztekkel legyen ellenőrizve. Ennek az az oka, hogy a feladat nagyrészét maga a játékmotor végzi. A repülésnél például, az általam írt kód kiszámolja, hogy az adott helyzetben milyen erők és forgatónyomatékok hatnak a repülőre, de azok alapján elmozdítani majd csak a fizika motor fogja.

Ennek ellenére azonban akadnak olyan kódrészek, melyekre illenek a unit tesztek. Ilyen például a chat szűrő logikája, ami kiszűri mind a vezérlési karaktereket, mind pedig az illetlen szavakat.

Továbbá többjátékos módban az adatok megosztására használt struktúra is tesztelve van, hogy megbizonyosodjak arról, hogy a klienseken pont azok az adatok fognak előállni, amelyek meg lettek osztva.

Ezen tesztekkel az imént említett funkcióknak a helyességéről már jóval az előtt meg tudtam bizonyosodni, hogy azokat éles adatbázison kipróbáltam volna.

8.3. Elosztott tesztelés

Ugyan saját magam rengeteget futtatom a játékot, a projekt mérete túl nagy ahhoz, hogy egy játékos kellő lefedettséggel kipróbálja a funkciókat. Emiatt az alkalmazás béta verzióját elérhetővé tettem a Google Play áruházon, mint előzetes hozzáférésű alkalmazás. Ennek az a célja, hogy a fejlesztő a hivatalos kiadás előtt is már kaphasson felhasználói visszajelzéseket.

Már több olyan hibára is fény derült, amelyekre én hosszas játszogatás után sem bukkantam rá. De a sok „béta tesztelőnek” hála ki tudtam javítani. Ilyen volt a köd, amit a játékos elmerülésekor kapcsolok be, hogy víz alatti hatást keltsen. Ez a köd bizonyos esetekben nem kapcsolt ki, amikor a játékos visszaállította a repülőjét a kiindulási állapotába.

További előnye, hogy a többjátékos módot is valós játékosokkal tudtam tesztelni, szinte mindenhonnan a világ körül. Még olyan dolgokra is következtetéseket tudok levonni, hogy például a repülő átfestő menü elég intuitív-e? Az, hogy a játékosok szinte mindegyike átfestett repülővel csatlakozik arról tanúskodik, hogy a repülőt személyre szabó menü kellőképpen érthető.

Továbbá a szerver terherbírását is tesztelni tudtam, mert már több olyan alkalom is volt, hogy egy játékszoba megtelt, és így sem volt megfigyelhető semmi kiesés vagy lelassulás.

8.4. Teljesítmény teszt

Fontos célkitűzés volt, hogy a játék az alacsonyabb teljesítményű készülékeken is játszható legyen.

A tesztelés során több készüléken is játszottam. Az eredményeket a 8.1-es táblázat tartalmazza. A teljesítmény a másodpercenkénti képkockák számaként van kifejezve. Három szituációt vizsgáltam. Ezeket a 8.1-es ábrán lehet végigkövetni: maximális grafikai beállítás (bal), fák nélkül (közép), fák nélkül és gyors felhőkkel (jobb).

8.1. ábra: Teszteléshez használt grafikai szintek

A táblázat az eszköz megnevezése mellett egy kategóriát is tartalmaz, ami erős becsléssel megmondja, hogy milyen a készülék teljesítménye, játszás szempontjából.

Gyártó	TCL	Huawei	Alcatel	Blackberry	Samsung	Sony
Model	BeyondTV	Mediapad T3	1s	Priv	Galaxy A51	Xperia XZ3
Kategória	alsó kezdő	kezdő	felső kezdő	alsó közép	közép	csúcs
Max grafika (FPS)	4	5	20	25	45	60
Fák nélkül (FPS)	15	30	60	45	60	60
Gyors felhők (FPS)	25	60	60	60	60	60

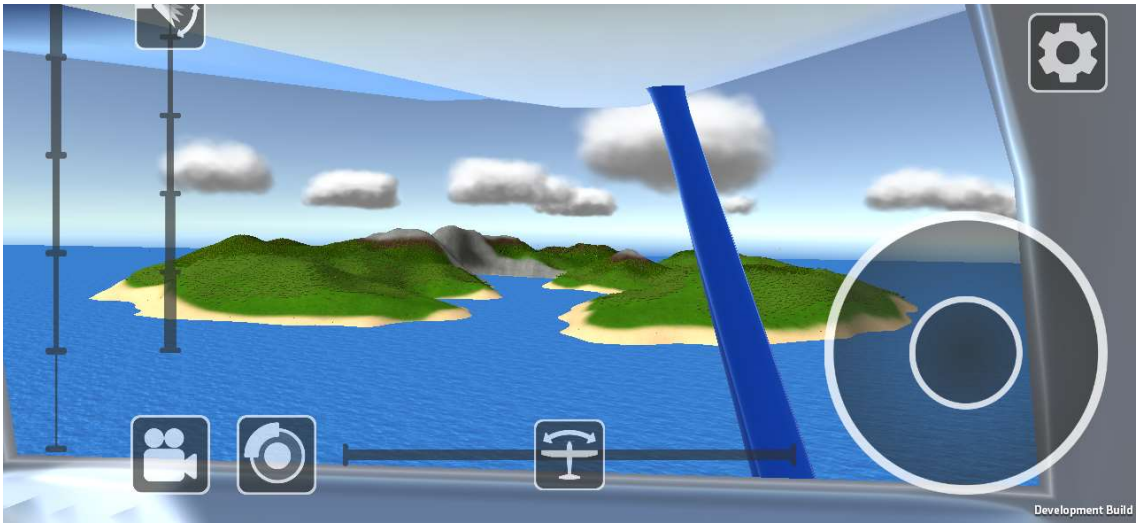
8.5. táblázat: A játék teljesítménye különböző eszközökön és grafikai beállításokon

A cél a 24 képkocka / másodperc sebesség elérése volt, minél alacsonyabb kategóriájú eszközökön is. A játékot egy androidos TV-n is futtattam, ami egyáltalán nem játékra lett tervezve, de még ezen is sikerült – minimális grafikai beállítások mellett – a 25 FPS elérése.

9. AZ ELKÉSZÜLT SZOFTVER BEMUTATÁSA

Egyjátékos mód

A szoftver alapvető funkciója az egyjátékos, azaz offline játék. A játékos ekkor szabadon repülhet a 9.1. ábrán látható szigeten, ami 4 darab repteret tartalmaz (két kicsi – két nagy).



9.1. ábra: A sziget

Ennek a játékmódnak nincsen különösebb célja, a játékos gyakorolhatja a leszállást, fejlesztheti a repülési képességeit, vagy, ha egy magával ragadóbb élményre vágyik, kipróbálhatja a virtuális valóság módot.

VR

Amennyiben a játékos készüléke tartalmaz gyro szenzort, és rendelkezik VR szemüveggel (pl.: Google Cradboard), akkor lehetősége van a játékot 3D-ben megtapasztalni, a virtuális valóság módot választva. (Lásd 9.2. ábra)

Ez nem csak annyiból áll, hogy a játékos már a mélységet is érzékelni fogja, hanem a gyro szenzornak hála, a játékbeli kamera leköveti a telefon mozgását, így a játékos akár körbe is nézhet a pilótafülkében.

Ebben a játékmódban is több opció van az irányításra. A játék kontrollerek mellett lehetőség van a repülőt egy másik eszközről is irányítani, egy erre a célra tervezett távirányító alkalmazással.

9.2. ábra: Virtuális valóság mód

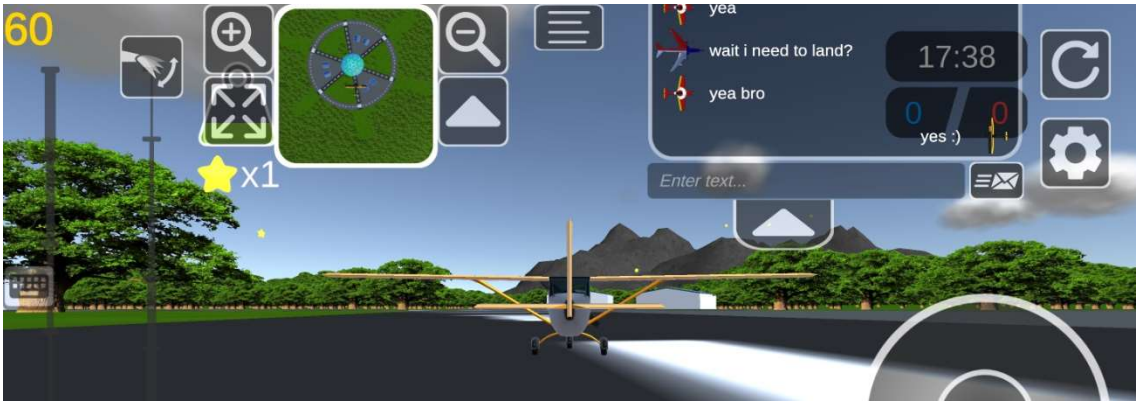
Multiplayer

A játék fő attrakciója a multiplayer, ahol a játékosok egy közös világban repülhetnek. Ez történhet lokális hálózaton, valamint az interneten. Az interneten történő játék esetében két játékmód érhető el: a szabad repülés (Free Flight) valamint a csillaggyűjtő (Star Collector).

A Free Flight mód nagyban megegyezik az offline játékkal, olyan értelemben, hogy konkrét küldetés nincsen, itt azonban pénzérmék lebegnek a sziget felett, amiket a játékos összegyűjthet, hogy növelje játékbeli egyenlegét. És találkozhat másokkal is, lásd 9.3. ábra.

9.3. ábra: Multiplayer

A Star Collector mód egy csapatos kompetitív játékmód, amiben nem céltalanul repülnek a játékosok, hanem két csapatban egymás ellen küzdenek. A feladat az, hogy minél több csillagot gyűjtsenek össze, és juttassák el a csapat bázisára, épségben. Ez a játékmód látható a 9.4. ábrán.



9.4. ábra: Csillaggyűjtő mód

Repülőfestő

Elsősorban a többjátékos mód érdekesebbé tételére, a repülők szabadon átszínezhetőek. Minden repülőhöz két matéria tartozik, amiknek a tulajdonságai (szín, fényesség, fémesség) felülírhatóak azzal, hogy a felhasználó egyedi matériákat definiál (lásd 9.5. ábra), amiket hozzárendelhet a repülőhöz. Így a játékos könnyen megkülönböztetheti magát az online játékban, akkor is, ha a többi játékos is azzal a repülő típussal játszik.

9.5. ábra: Repülő átfestése

10. ÖNÉRTÉKELÉS

Multiplayer

Amikor a projektet elkezdtem, a célkitűzésem az volt, hogy egy olyan repülés szimulátort hozzak létre, ami lehetőséget ad a játékosoknak arra, hogy közösen tudjanak játszani egy világban, mindezt térítésmentesen.

Ezen sor írásakor boldogan jelentem ki, hogy ez teljes mértékben sikerült.

Először is, a játék támogatja a multiplayert lokális hálózaton keresztül. Ezt a játékmódot a visszajelzések alapján elsősorban testvérek használják, akik így az otthoni hálózatukon keresztül tudnak közösen játszani, úgy, hogy arra mások kívülről nem tudnak csatlakozni.

A hátrány viszont az, hogy egy azon hálózatra kell csatlakoznia mindegyik eszköznek. Így egy baráti társaság például, nem tud egymással játszani, ha nincsenek fizikailag ugyanott (feltételezve, hogy nem használnak VPN-t). Ekkor válik szükségessé a világhálón történő játék.

A multiplayer megvalósítások közül mindenképpen ez volt a nagyobb falat, hiszen itt már egy dedikált kiszolgálóra is szükség van, aminek, továbbá költséghatékonynak is kell lennie, hogy a célkitűzésem „térítésmentes” része kivitelezhető legyen.

A kiszolgáló szerepét egy adatbázis szerver látja el, ami alacsony anyagi ráfordítást igényel. Cserébe azonban a kliensek közötti átvitel lassabb, mivel a szerver nem továbbítja rögtön a kapott adatokat, hanem csak a kliensek kérésére. Emiatt ritkább az adatcsere, mint lokális többjátékos mód esetében. Mindezek ellenére azonban élvezhető a játék a saját véleményem szerint is, és a Google Play visszajelzések alapján is.

Online játékban a hagyományos szabadon repülős „Free Flight” mód mellett egy exkluzív csillaggyűjtő játékmódot is ki lehet próbálni, amiben nem céltalanul repülnek a játékosok, hanem két csapatban egymás ellen küzdenek. A tesztelés során mindent rendben találtam, de a jelenleg még szűk játékosbázis miatt ez a játékmód többnyire kihasználatlan. Azonban belefutottam már játékosokba itt is, mint ahogy az a 9.4-es ábrán is látható volt.

Grafika

Már a projekt kezdetén világos volt számomra, hogy a grafika realisztikussága terén labdába sem rúgok a nagy cégek (például Laminar Research) mellett, akik külön grafikusokat dolgoztatnak. Ezért inkább a játékélményre fókuszáltam.

A konkurensek szimulátoraiban a grafika szempontjából leginkább a borongós megvilágítás zavart. Ezért a projektemben minél derűsebb hangulatot igyekeztem elérni az erős megvilágítással és a világos, szinte már túlszaturált színekkel.

Grafika terén, amire különösen büszke vagyok, az az erdők megvalósítása. Amikor még a beépített erdőgeneráló eszközzel dolgoztam, a Blackberry Priv készülékemen megközelítőleg egy képkockát tudott a játék másodpercenként kirenderelni. Ez a szám a

saját megoldásommal 25-re nőtt, miközben az erdők csak gyarapodtak. Az erdők méretének érzékeltetése végett, a 10.2. ábrán látható szigeten pontosan 56 659 darab fa található. Mivel minden fa 4 háromszögből áll, ez összesen 226 636 háromszög, amit a játéknak ki kell rajzolnia.

10.1. ábra: Rengeteg erdő

A sok-sok fával azt akartam elérni, hogy a terep ne tűnjön kopárnak, hanem távolról nézve szép, egybefüggő erdőket alkossanak. A plakátfák közelről nézve természetesen egyáltalán nem tűnnek realisztikusnak, azonban a projektem abban a kiváltságos helyzetben van, hogy mivel repülőgép szimulátor, a játékosok messze a fák felett fognak tartózkodni a játékkal töltött idő szinte egészében. Emiatt elég volt az erdők távoli látványára fókuszálni.

A felhők is hasonlóan jól sikerültek. Eleinte nem gondoltam túl ezeket. Blenderben készítettem egy modellt több megnyújtott gömbből összerakva, aminek lelaposítottam az alját, és ezt használtam felhőnek. Ezek láthatóak a 10.3-as ábrán.

10.2. ábra: A régi felhők

Ezek a felhők - véleményem szerint - beleillettek a játék stilizált stílusába. A visszajelzések azonban világosak voltak: A felhők nevetségesek, le kell őket cserélni.

Végül az erdőknél megszerzett tapasztalataimat használtam fel, hogy szofisztikáltabb felhőket alkossak, amik már kellően realisztikus hatást nyújtanak, mint ahogy az a 10.4-es ábrán látható. A látvány mellett azért is jelentettek az új felhők nagy javulást, mert

most már át is lehetet rajtuk repülni és pont azt a ködös hatást keltik, amire az ember számít. Lásd 10.5-ös ábra.

10.3. ábra: Az új felhők látványa

10.4. ábra: Repülés egy felhőben

A repülők

A játékban jelenleg hét különböző repülővel lehet játszani (lásd 10.6. ábra). Ez nem éri el a konkurensek szimulátoraiban lévő repülők számát, de ez nem is volt célom, tudván, hogy ez egy egyszemélyes projekt.

10.5. ábra: Repülőgép típusok

Ezek közül csak az elsővel rendelkezik a játékos eleinte, majd, ahogyan játékbeli pénzt gyűjt, a többi repülőgépet is feloldhatja. Az összes modellre igaz, hogy rendelkezik pilótafülkével, a műszerek működnek, és a kormányfelületek mozognak. A fejlettebb repülőgépek a futóműveiket is be tudják húzni.

A gépeket át is lehet festeni. Ez a funkció elsősorban azt a célt szolgálja, hogy a játékosok megkülönböztethessék magukat amikor egymással játszanak. A tapasztalat az, hogy a játékosoknak valóban van erre igényük, mivel alig lehet látni repülőket az online játékban, amelyek az alapértelmezett matériákat használják.

Irányítás

Az irányítás több módon is történhet, hogy mindenki megtalálja azt az opciót, ami neki a legkézenfekvőbb. A legtöbb esetben a gyorsulásmérős vagy virtuális joystick mód az ideális. Ezek azonban nem működnek akkor, ha a VR mód aktív, ugyanis a UI elemeket akkor nem lehet kezelni (mert az eszköz az ember fején van). Erre az esetre is vannak opciók. Lehetőség van egy- illetve kétkezes játékkontrollert használni, továbbá billentyűzetet, sőt, egy külön kontroller alkalmazással is irányítható, egy másik androidos eszközről.

Ez utóbbi egy másik olyan része a projektnek, ami nagy újítás a konkurensek szimulátoraihoz viszonyítva. A kontroller alkalmazás legnagyobb előnye, hogy mivel a gyorsulásérzékelőt használja botkormány bemenetként, ezért sokkal finomabb irányítás és precízebb manőverezés érhető el. Ezen felül, a felhasználói felület pont a szimulátorhoz lett személyre szabva. Így például a gázkart nem két gombbal kell állítani (egyikkel gázt adni, másikkal levenni), hanem egy sokkal intuitívabb csúszkával. (Az alkalmazás felülete a 7.16-os ábrán látható.)

Az egyetlen nehézséget az jelentheti, hogy a fizikai gombokkal ellentétben, a képernyőn nem lehet a UI elemeket kitapogatni, miközben látni sem látja a játékos, mert éppen a fején van egy VR szemüveg. De, mint ahogy azt a megvalósítás részben is említettem, a gombok és csúszkák nagyok, és könnyen megjegyezhető helyen vannak. Néhány másodperc memorizálást követően, az elemek könnyen eltalálhatóak nézés nélkül is.

A VR mód mellett, ez az irányítási mód akkor is hasznos, ha az ember nagy képernyőn (például egy androidos TV-n) játszik, különösen akkor, ha az nem támogat érintéses bemenetet.

VR

Az irányítás részben már szóba került a VR. A funkció hibátlanul működik. A megvalósítás fejezetben ismertetett módon előállított két kép térhatást nyújt, és a készülék giroszkópjának köszönhetően a kamera leköveti a játékos fejének mozgását, aki így körül tud nézni, mintha tényleg a repülő pilótafülkéjében ülne.

A VR működése a 9.2-es ábrán látható. Ha az olvasó fedésbe tudja hozni a kép jobb és bal oldalát, oly módon, hogy a szemeivel az ábra mögé fókuszál, akkor a kettő képből három lesz, és a középső azt a térhatást nyújtja, amit a játékos VR-ban tapasztal.

A funkció szintén nagy népszerűségnek örvend, azonban a célpiaca jelentősen kisebb a játék célpiacánál, ugyanis nem minden játékos rendelkezik VR szemüveggel, vagy ahhoz tartozó kontrollerrel. A visszajelzésekből azt a következtetést vonhatom le, hogy akik megtehetik, azok előszeretettel játszanak ebben a módban. Nagycsaládi összejöveteleken is ez a személyes tapasztalatom, ugyanis a gyerekek elsősorban VR-ban szeretnek játszani.

Az egyetlen hátrány az, hogy a chat funkció nem használható. A játékmódokhoz szükséges UI elemek és a kistérkép elérhető itt is, de a chat azonban nem. Ennek az az oka, hogy a játékos nem tud szöveget begépelni. Persze a chat ettől még megjelenhetne, úgy is, ha a felhasználó nem tud üzenetet küldeni, de az a félő, hogy annak ténye, hogy nem tud például egy feltett kérdésre válaszolni akkora frusztrációhoz vezet, ami negatívan hat a játékról, és különösen a VR módról alkotott véleményére. Ezért nem jelenik itt meg a chat semmilyen formában sem.

11. TOVÁBBFEJLESZTÉSI LEHETŐSÉGEK

Több repülőgép

Egy egyértelmű továbbfejlesztési lehetőség az, hogy több repülőgép típussal lehessen játszani. A jelenlegi hét repülőt fel szeretném növelni tízre a játék hivatalos kiadásáig.

E mellett más közlekedési eszközökkel is tervezem kibővíteni, például helikopterekkel, vagy akár UFO-kkal is, hogy ezek változatosabbá tegyék a játékot.

Több sziget

A kezdeti sziget keresztbe történő átrepülése bő hat percet vesz igénybe a kezdeti repülőgéppel. A körberepülése pedig megközelítőleg 17 perc. A nagyobb repülőgépek hozzáadásával viszont ez a sziget már nagyon gyorsan bejárható, így a játék hamar unalmassá válhat. Új szigetek hozzáadásával viszont sokkal több időbe fog telni, míg a játékos mindent felfedez, akkor is, ha nagy repülőt használ.

Több játékmód

A tesztelők felől sokszor kapok olyan visszajelzéseket, hogy egy harci játékmódot szeretnének látni, ahol kedvükre lövöldözhetnek. Határozottan tervezek még játékmódokat hozzáadni, jelenleg azonban a játékosok hiánya okozza a legnagyobb problémát az online játékban. Azzal, hogy új módokat adok hozzá, a játékosokat még inkább lecsökkenteném az egyes játékmódokban, így kisebb valószínűséggel fog két játékos belefutni egymásba. Emiatt inkább későbbre halasztom ezt a továbbfejlesztési lehetőséget.

Minták festése a repülőgépekre

Az értékelés fejezetben már említettem, a játékosok legnagyobb része él a lehetőséggel, hogy a gépét egyedivé tegye azzal, hogy átszínezi. Az egyedivé tételt azonban limitálja az, hogy csak homogén matériákat lehet definiálni. Jó lenne, ha lehetne mintákat is ráfesteni a repülőgépekre.

A feladat nehézsége abban áll, hogy ezeket a matériákat át is kell küldeni a hálózaton, mivel az egész funkció a többjátékos mód miatt jött létre. Emiatt semmiképen sem lehet bitmap képeket generálni, mert az jelentős adatforgalom növekedéssel járna. Egy lehetséges megoldás, hogy vektorgrafikához hasonlóan, alakzatok egyenletét adja meg a játékos, és a repülőgépet kirajzoló shader minden pixelre eldönti, hogy metszi-e az alakzatot. És ha igen, akkor az alapértelmezett szín helyett az alakzat színét jelenítené meg.

Nem-játékos karakterek

Az egyjátékos módot érdekesebbé tenné, ha lennének NPC-k, azaz program által irányított repülők. Ezek egy előre maghatározott útvonalon repülnének, reptértől reptérig. A valódi játékos ezekkel versenyt tud repülni, beléjük tud csapódni, esetleg megpróbálhat

leszállni egy nagy repülő szárnyára, és még további kreatív módokon tudná szórakoztatni magát.

Ezeket a multiplayer módban is fel tudom használni arra, hogy a szerver ürességét minimalizáljam. A kihívás itt az, hogy az NPC-k mozgását szinkronizáljam a kliensek között.

Szél

Légmozgásokat is implementálni tervezek. Ezek a lágy szellőktől a viharos széllökésekig terjednének. Ez a funkció elsősorban az ügyesebb játékosok részére szól, akik szélcsendben már tudnak repülni és le is tudnak szállni.

A nehezebb kérdés itt is a többjátékos mód. Amíg az ember egymagában játszik offline, kedvére állíthatná a szélerősséget. Online azonban nem állíthatja mindenki tetszőlegesen, hanem a szervernek kell megmondania, hogy mekkora és milyen irányú a szél. Így viszont találni kell egy olyan szél beállítást, ami a legtöbb játékosnak megfelel, különben az negatívan hat a játékosok számára.

12. TARTALMI ÖSSZEFOGLALÓ

Ebben a dolgozatban egy androidos eszközöket célzó repülés szimulátor elkészítéséről olvashatunk. A projekt elsődleges célja, hogy ingyenes online többjátékos szolgáltatást nyújtson. Ebből az következik, hogy a projektnek költséghatékonynak kell lennie.

A játék központi eleme az online multiplayer, amihez szükség van egy szerverre. Ennek, az általam talált legköltséghatékonyabb megvalósítása egy adatbázis szerverrel lehetséges, amit üzenőfalnak használnak a kliensek. Az adatokat védendő, egy PHP biztonsági kapu lett felállítva az adatbázis elé, ami ellenőrzi és szűri a forgalmat.

Lehetőséget adok arra, hogy a játékosok személyre szabják a repülőiket azzal, hogy átszínezik azokat. Ezzel megkülönböztethetőek lesznek, miközben együtt játszanak. Továbbá, elérhető egy csapatos, kompetitív játékmód, ahol a játékosok összemérhetik repülési képességeiket.

Mindemellett a virtuális valóságot is támogatja a játék. Ehhez szükség van egy giroszkópos mobilra, egy VR szemüvegre, és egy játék kontrollerre. Ez utóbbi egy másik androidos eszköz is lehet, amire telepítve van az e célra fejlesztett kontroller alkalmazás.

A játékot, mint előzetes hozzáférésű alkalmazást, publikáltam a Google Play áruházon. A felhasználók tesztelői visszajelzéseket küldhetnek, amelyek segítségével rejtett hibákat is ki tudok javítani. Ezekből a visszajelzésekből továbbá elmondható, hogy a játékosok többségének tetszik a játék.

A projektet a hivatalos kiadás előtt még bővíteni tervezem. Először is több repülőgépmodellt teszek elérhetővé, és új szigetekkel teszem érdekesebbé a terepet. Ezek mellett kiegészítem a személyre-szabás funkciót azzal, hogy mintákat is lehessen a gépekre festeni. Bevezetek még program által irányított repülőket is, amelyek izgalmasabbá teszik majd az egyedüli játékot.

13. SUMMARY

In this thesis, we read about the creation of a flight simulator, targeting android devices. My goal with the project is to offer free multiplayer service to the players. Because of this, it is essential for the project to minimize the costs.

The project is centred around the online multiplayer, which requires a dedicated server to operate. The most cost-efficient solution I could come up with, is to use a database server, which the clients can use like a message board. In order to protect this database, a PHP security gate was erected, which supervises and filters the network traffic.

I let the players customize their planes, by changing their materials. This way, they can differentiate themselves when playing together. Also, there is a team based competitive game mode, in which players can prove their flying skills.

The game also offers the option of virtual reality. All that's needed for this, is a phone with a gyro sensor, VR goggles, and a controller. The alter could also be another android device, running a controller application, specifically made for this purpose.

I've published my game as an early access application on Google Play. Here, players can provide testing feedback, helping me fix issues, which I otherwise wouldn't have found out about. Also, I can see that the majority of players enjoy the game, from the feedback I'm getting.

Before the official release, I'm planning to further expand the game, by adding more planes and new islands, to make the map more interesting. I'm also improving the customization system, by letting players paint shapes on their planes. Furthermore, I'm introducing non-player characters, aimed at making single player more exciting.

14. ÁBRAJEGYZÉK

15. TÁBLÁZAT JEGYZÉK

16. IRODALOMJEGYZÉK

[1] X-Plane 9. *Laminar Research*, (https://play.google.com/store/apps/details?id=com.laminarresearch.xplane_default), utoljára megtekintve: 2021.05.06.

[2] X-Plane Flight Simulator. *Laminar Research*, (https://play.google.com/store/apps/details?id=com.laminarresearch.x_plane10), utoljára megtekintve: 2021.05.06.

[3] Infinite Flight – Flight Simulator. *Infinite Flight LLC*, (https://play.google.com/store/apps/details?id=com.fds.infiniteflight), utoljára megtekintve: 2021.05.06.

[4] Flight Pilot Simulator 3D Free. *Fun Games For Free*, (https://play.google.com/store/apps/details?id=com.fungames.flightpilot), utoljára megtekintve: 2021.05.06.

[5] A Google Play díja, (https://support.google.com/googleplay/android-developer/answer/6112435), utoljára megtekintve: 2022.03.30

[6] Az AppStore éves díja, (https://developer.apple.com/support/compare-memberships/), utoljára megtekintve: 2022.03.30

[7] Az Android Studio a hivatalos android IDE, (https://developer.android.com/studio/features), utoljára megtekintve: 2022.04.02

[8] Amazon árajánlat, (https://calculator.aws/#/createCalculator/EC2), utoljára megtekintve: 2022.03.30

[9] Fábián, A.: PPL kézikönyv A repülőgép-vezetés elmélete. SKYLIGHT CREATIVE EC., Budapest 2010, pp. 23-27.

[10] Brian J. Cantwell: „AA200 Applied Aerodynamics”, STANFORD EGYETEM, 2014, 12. fejezet, pp. 23-25. (https://web.stanford.edu/~cantwell/AA200_Course_Material/AA200_Course_Notes/AA200_Ch_12_Wings_of_Finite_Span_Cantwell.pdf), utoljára megtekintve: 2021.10.25