# **Záródolgozat – dokumentáció**

**Labirintus Játék**

Kirsch János Márk

Tartalom

[Alapkoncepció: 2](#_Toc5026427)

[Témaválasztás 2](#_Toc5026428)

[Kiinduló rendszer összefüggései: 3](#_Toc5026429)

[Rendszerigény: 5](#_Toc5026430)

[Rendszertervezés: 5](#_Toc5026431)

[Web: 8](#_Toc5026432)

[Funkció bemutatása: E-mail cím módosítása 9](#_Toc5026433)

[A felhasználó: Tipikus futtatási példák 11](#_Toc5026434)

[Biztonsági kérdések: 12](#_Toc5026435)

[Mit tehetek, ha mégis én magam szeretném a jelszavakat tárolni? 12](#_Toc5026436)

[Mi lehet akkor a legjobb saját megoldás jelszavak titkosítására? 12](#_Toc5026437)

[Adatbázistervezés: 13](#_Toc5026438)

[Ergonómia: 14](#_Toc5026439)

[Néhány szó az objektum orientált programozásról (OOP) az Alkalmazásomban: 14](#_Toc5026440)

[Tesztelés: 14](#_Toc5026441)

[Tesztelés a programomban: 14](#_Toc5026442)

[Teszt 1: 15](#_Toc5026443)

[Teszt 2: 15](#_Toc5026444)

[Komplex alkalmazás: Összegzés: 16](#_Toc5026445)

[Eredeti tervek: 16](#_Toc5026446)

[Web: 16](#_Toc5026447)

[Adatbázis: 16](#_Toc5026448)

[Végleges verzió: 17](#_Toc5026449)

[Asztali: 17](#_Toc5026450)

[Web: 17](#_Toc5026451)

[Adatbázis: 17](#_Toc5026452)

[Fejlesztési lehetőségek: 18](#_Toc5026453)

[Fontosabb akadályok a fejlesztés során: 18](#_Toc5026454)

[Fejlődéseim: 19](#_Toc5026455)

[**Köszönetnyilvánítás** 21](#_Toc5026456)

[Szakdolgozat írásához felhasznált források: 21](#_Toc5026457)

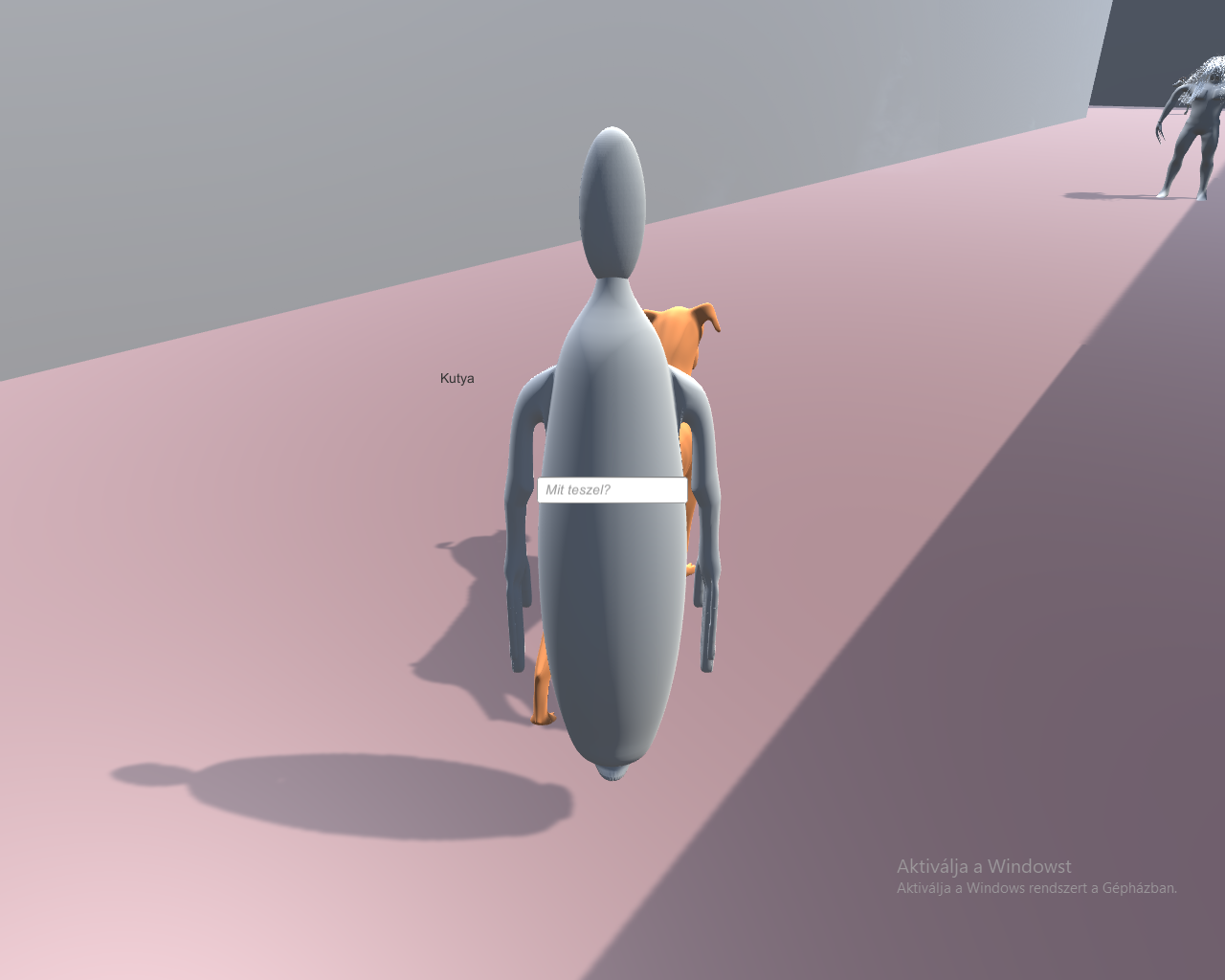
[Plágium - Nyilatkozat 22](#_Toc5026458)

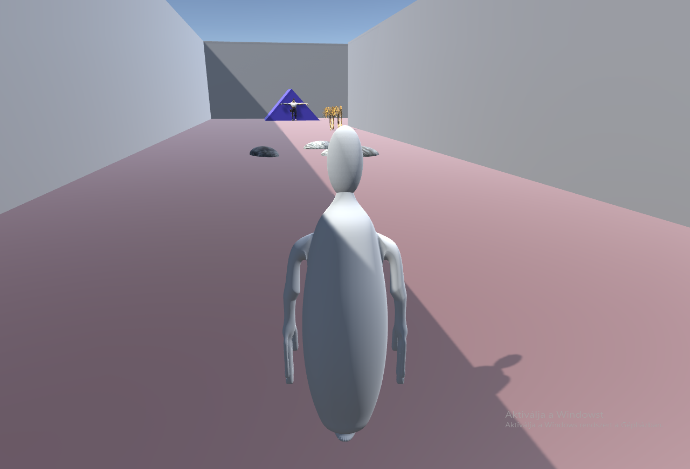
# Alapkoncepció:

## Témaválasztás

A záródolgozat témájának háromdimenziós játék elkészítését választottam, mely téma gyerekkorom óta lenyűgöző és távoli számomra, illetve olyan téma, ami későbbi munkám során referenciaként és tapasztalat formájában is előnyökkel kecsegtethet.

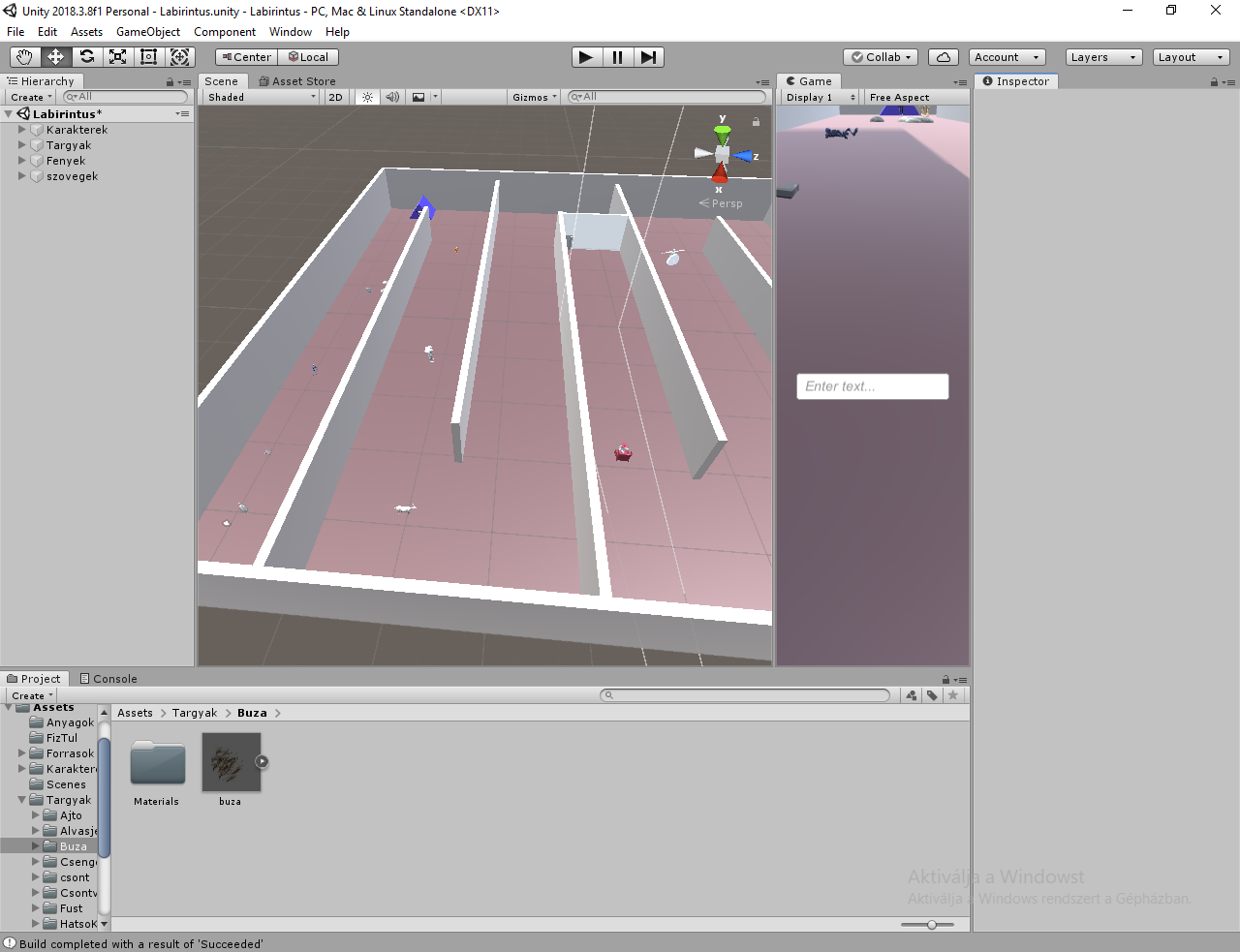
Az asztali alkalmazásnak a magja egy labirintus játék melyben a karakter mozogva, más objektumokkal interakcióba lépve, szövegben beírt parancsok által a pálya elemeit kreatívan felhasználva juthat tovább az akadályokon. A játék célja végső soron, hogy a játékos átérjen a túloldalra és ott kijusson az útvesztőből az utolsó akadályt is leküzdve.





*képernyőkép: Labirintus képernyőkép: interakció karakterrel*

Számos játék készül az általam használt program segítségével. Vannak nagy cégek, akik az általuk készült alkalmazással világszínvonalú sikereket értek el és a programjuk még az E-sportok között is tiszteletre méltó helyet vívott ki magának – Blizzard, Hearthstone. Ám vannak egyéni fejlesztők is, akik nagy sikereket érnek el, vagy éppen csak néhány ember szórakoztatását biztosítják közeli körben.

Mivel egy játékos több játékkal is nyugodtan játszhat a verseny nem olyan erőteljes, mindenki érvényesülhet valamilyen szinten, ha tetszetős az általa elkészített termék. 

*kép: Unity 3D Engine*

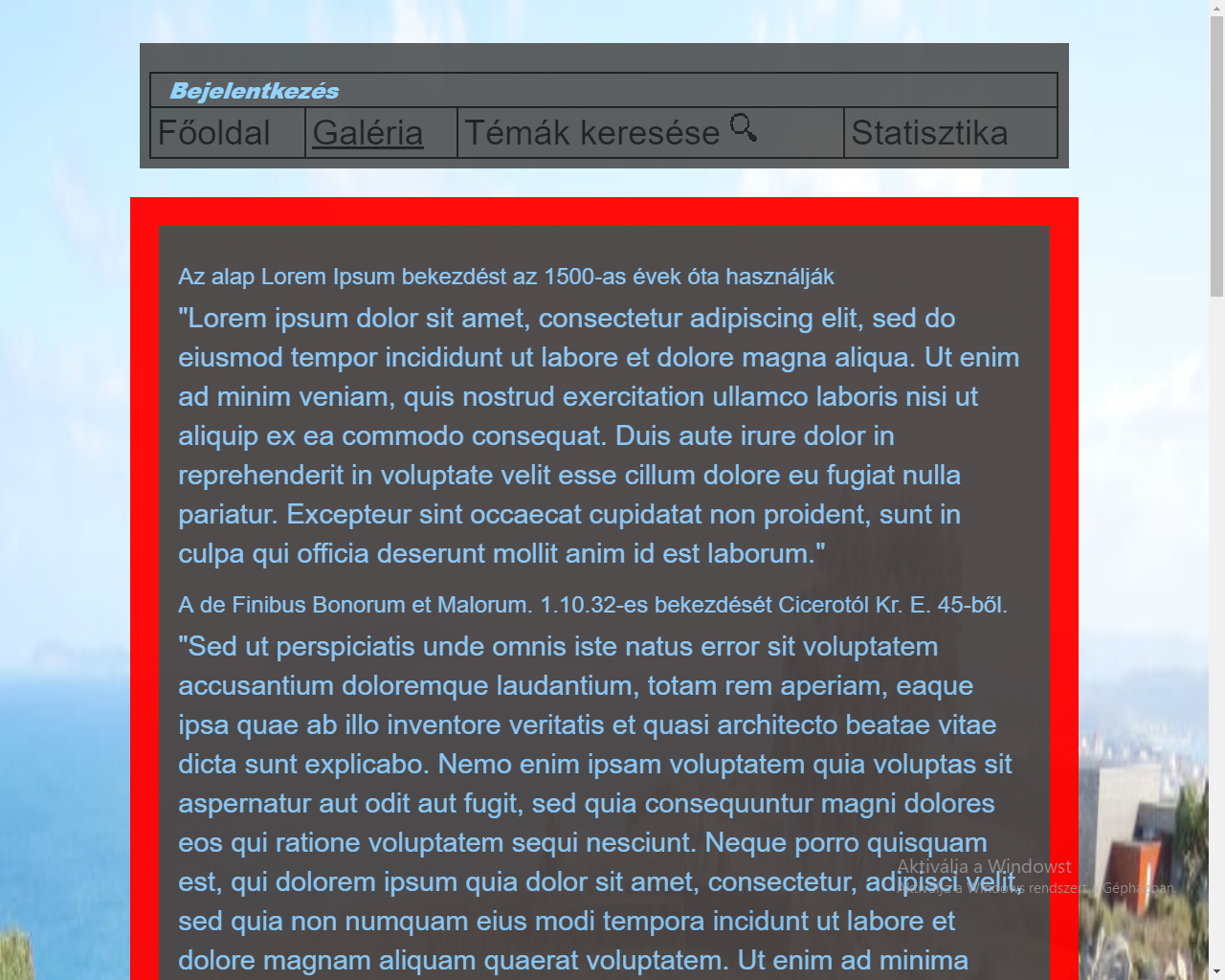
## Kiinduló rendszer összefüggései:

Az asztali alkalmazás maga egy menürendszer, mely a játék megnyitásához vezet. Az adatbázis első sorban a felhasználók adatait tárolja, amik azonosításhoz és online játékhoz szükségesek, illetve szerverekét, régiókét.



*kép: Asztali alkalmazás, főmenü*

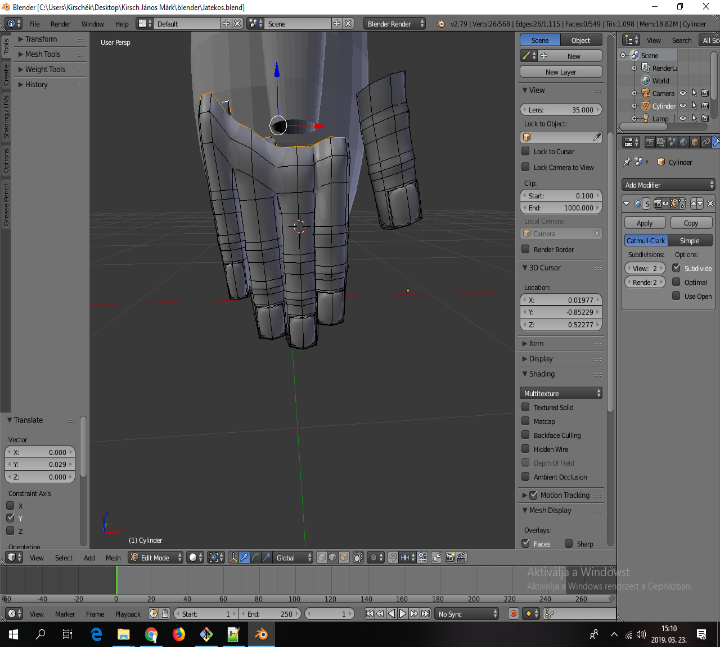
A komplex alkalmazáshoz tartozó weboldal egy fórum ahol a játékosok témákat tárgyalhatnak, képeket tölthetnek fel és tekinthetnek meg, az adminisztrátorok a főoldalon pedig információkat oszthatnak meg velük. Lehetőség van a felhasználók számára ellenőrzött úton az adatok módosítására.

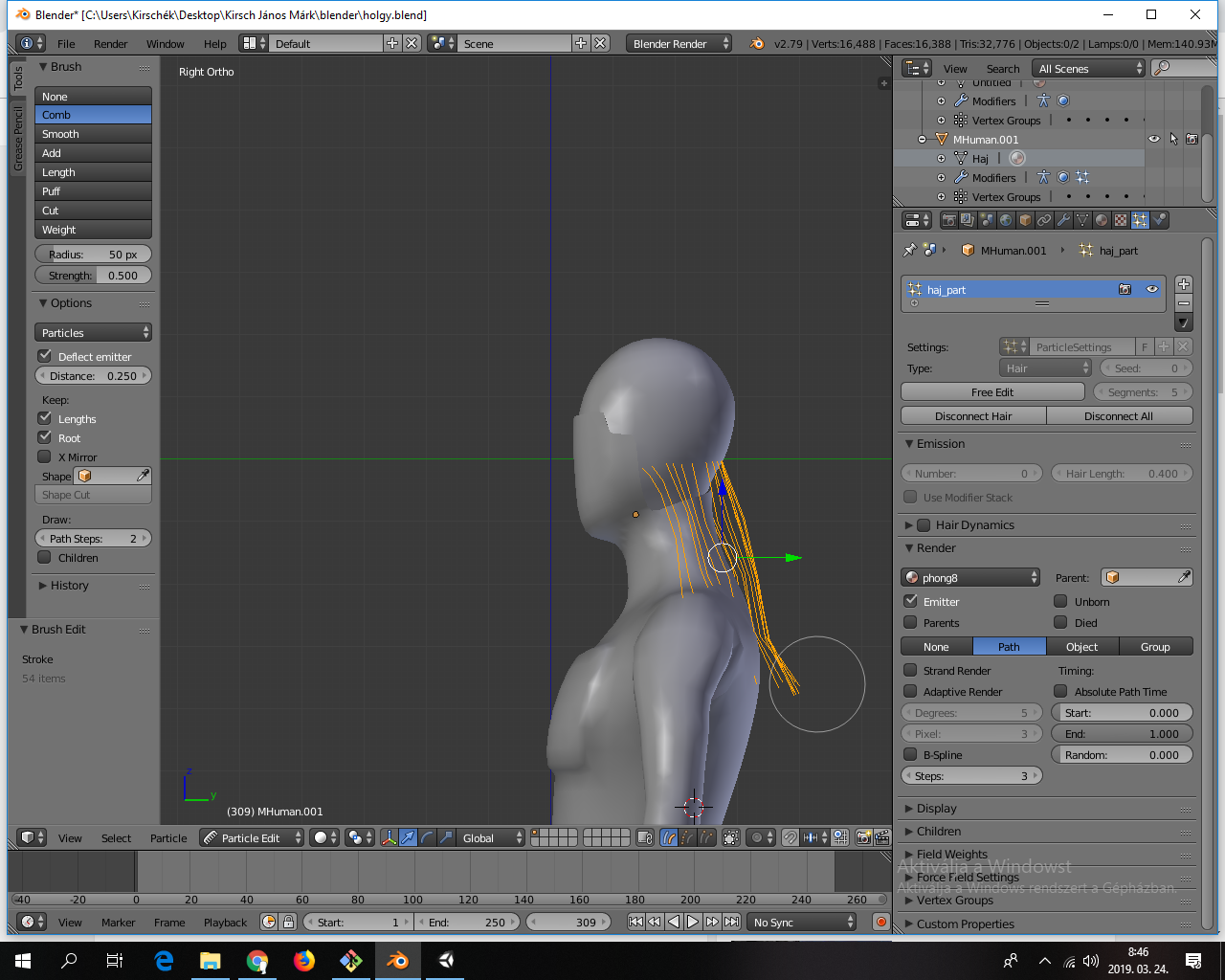


*képernyőkép: főoldal bejelentkezés nélkül*

Kiválasztott megoldások:

#### A grafikai elemek kinézetét blenderben készítettem, interneten megtalálható kész anyagok felhasználásával, azokat módosítva, illetve saját magam által készítettek is akadnak közöttük.





*képek:: Szerkeszttés Blender Studio Alalmazással*

#### Az első két dolog, amellyel foglalkoznom kellett a játékos mozgása illetve a kamera kezelése.

#### A játékos y tengelyen az egér pozíciójának megfelelően, arányosan elmozdul – így az egér jobbra-balra mozgásának megfelelően a játékos karaktere is elfordul, miközben a kamera végig a játékosnak ugyanarra a pontjára fókuszál.

#### 

#### A játékos egy előre elkészített függvény segítségével a forgatásának megfelelő irányhoz képest elmozdítható. Előre hátra gyorsulással, lassulással, megállással mozgatható (ezt saját metódusok által értem el), illetve ezekre való tekintet nélkül jobbra-balra erőhatás szimulálható elérve a megfelelő jóleső kontrollálását a szereplőnek. Ám a megszokottól eltérően a nagy helyre való tekintettel az én játékomban nem szükséges lenyomva tartani az előre – hátra iránygombokat, a karakter egyszeri megnyomás után önmagától megy, ellenkező irányba mozgással lassítható, valamint az alt lenyomva tartásának hatására „sprintelés” azaz nagyobb sebesség elérése lehetséges.

#### Mozgás közben a kamera végig a játékosra néz, ehhez a fajta irányításhoz én kifejezetten kedvelem a következő megoldást:

#### Míg a kamera együtt mozog a játékos forgásával, úgymond „kering” körülötte addig az egér föl-le mozgatásával a nézet is változik jobban megfigyelhetővé téve a játékteret.

# Rendszerigény:

#### Nincs túlságosan nagy teljesítményre szükség, a felbontés és a minőség játék elindítás előtt módosítható így bőven elég egy egyszerű számítógép is két magos Celeron processzorral 1,6 GHz-es órajellel, HDD merevlemezen 1,4-2GB tárhellyel, 2GB RAM-mal, windows 2000 vagy annál újabb operációs rendszeren, jelen állapotában azonban csak számítógépeken működik.

# Rendszertervezés:

### Asztali:

Követelményfeltárás:

#### Újszerű élvezhető kreativitással dolgozó játék

#### Labirintus, megfelelően működő fizika

#### Jó irányíthatóság

#### Képernyőn a történések legyenek egyértelműek a felhasználó számára

A rendszer felépítése:

#### A felhasználó elindítja a programot, amely magának a játéknak a „kezelőfelülete”, ahol kiválaszthatja a játék módját, korábbi mentéseket elérhet, beállításokat végezhet, bejelentkezhet.

Web:

#### Lehetőség bejelentkezéssel és anélkül az oldal látogatására

#### Képernyőképek megtekintése nagy felbontásban

#### Hely biztosítása a felhasználók és fejlesztők illetve a felhasználó és felhasználó között kommunikációhoz

#### Ellenőrzés után fiókadatok módosítása

#### Kijelentkezés

Szoftver-architektúra:

#### Mi a szoftver-architektúra (vagy rendszer architektúra)?

#### Az architektúra szó egy metafora, mely az épületek szerkezetével, architektúrájával azonosítja a rendszereknek hasonló modelljét

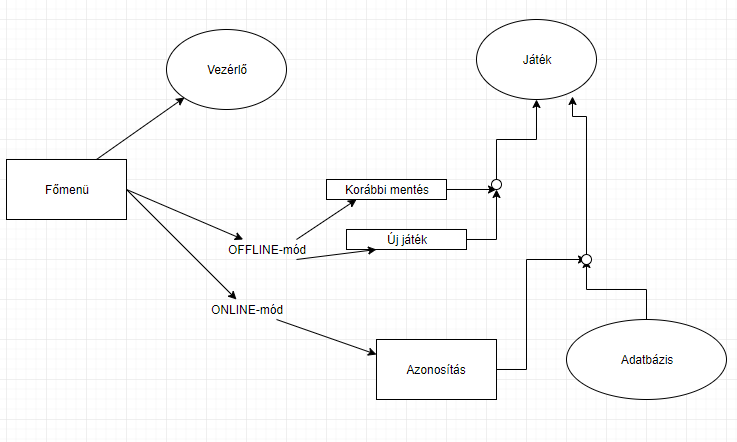
#### A rendszert felépítő alrendszereket összefogó rendszer formula

#### Alrendszerek lényegi megtervezése és azok tulajdonságai

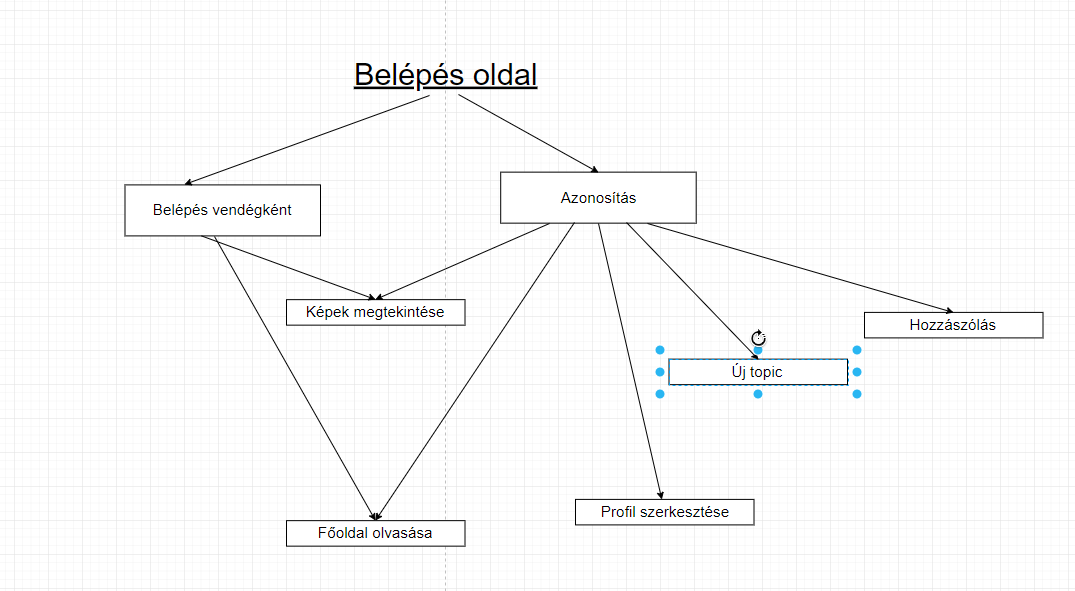
#### Vezérlés, valamint az alrendszerek közötti kommunikáció struktúrája

#### Az alapvető felépítési döntéseket fekteti le, melyek megváltoztatása költséges

Asztali:



Web:



*az ábrák draw.io segítségével készültek*

Szoftverkövetelmények:

#### Amikor a szoftverek minőségi követelményeiről beszélünk az első, amit mindenképpen szükséges megemlítenünk az ISO/IEC 9126-os szabánycsoport. Tartalma hat fő szempontot követ, ezek pedig:

#### 

#### Felhasználhatóság: A program megérhető és megtanulható legyen, könnyítse meg a felhasználó életét, ha bizonyos körülmények között használja azt

#### Funkcionalitás: Megfelelően végezze feladatát

#### Fenntarthatóság: Képesnek kell lennünk a rendszer elemeinek folyamatos tesztelésére, hajtsa végre stabilan rendeltetését

#### Hatékonyság: Erőforrások megfelelő felhasználása

#### Megfelelő idő alatt teljesítse a kéréseket

#### Megbízhatóság: Hibatűrés (például: hibás adatok bevitele esetén a rendszer megfelelően működik tovább)

#### Lehetőség a visszaállításra

#### Programom elkészítése során törekszem ezek betartására az első kész verzió bemutatása előtt.

Funkciók bemutatása:

1. Asztali:

#### A program elindításakor (3dLabirintus.exe) egy windows form standard ablak ugrik fel, mely tartalmazza a következő lehetőségeket:

#### Játék indítása

#### Online-mód

#### (Beállítások)

#### Kilépés

1. Játék elindítása

#### Megnyílik egy új windows form ablak a következő gombokkal

(windows ablakkeret gombjain kívül):

#### Új játék

#### Betöltés A mentett játékhoz való visszatérési lehetőség helye

#### Vissza

#### Az új játék gomb megnyomására megnyílik egy ablak, mely a játék konfigurációs lehetőségeit tartalmazza

#### A graphics fül alatt beállítható a felbontás illetve, hogy ablakos, vagy teljes képernyős módban fusson a játék

#### A grafika minősége nagyon alacsonytól ultráig összesen hat lehetőséggel, mely elősegíti az alacsonyabb erőforrású eszközökön való futtatást a minőség megőrzése mellett

#### Illetve lehetőség van monitor kiválasztására, melyik legyen az amelyiken a játék fusson

#### Az input fülnek jelentősége egyelőre nincs, ha a konzolon való futáshoz a szoftveres háttér elkészül, értelmet nyer

#### Play! gombra a játék elindul

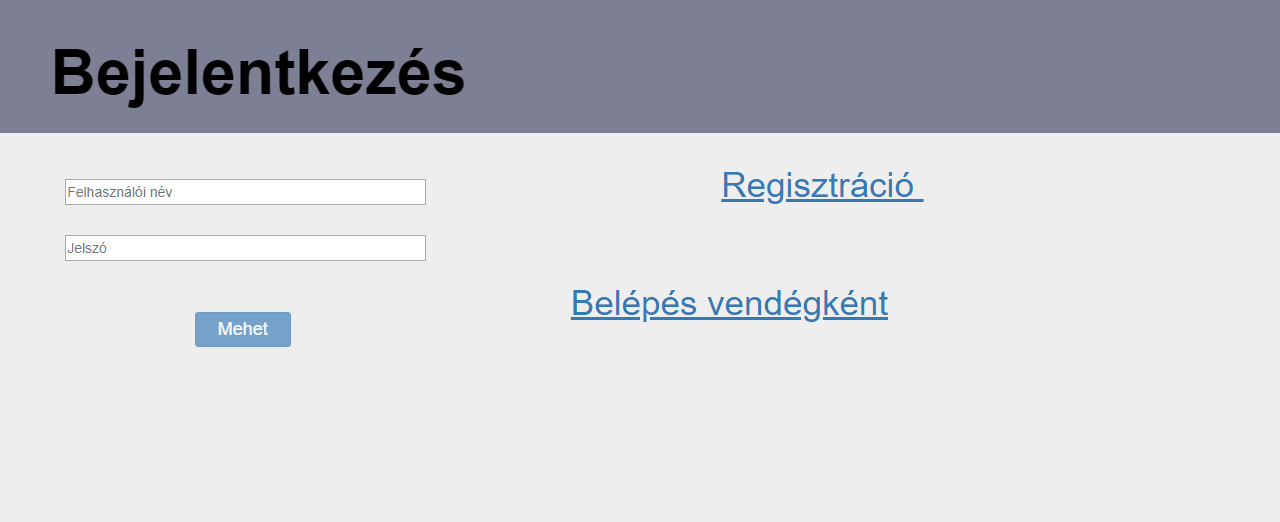
#### Quittel pedig visszatérhetünk a fő rendszerbe

Online-mód: Azonosítás után adatbázis segítségével játék több emberrel az interneten keresztül (ez még nem készült el!)

Beállítások: Hang és fényerő helye

Web:

Az index.php, vagyis az URL beírása utáni első ablak egy belépési űrlaphoz vezet, ahol azonosítás után vagy anélkül beléphetünk, a főoldalra illetve regisztrálhatunk.



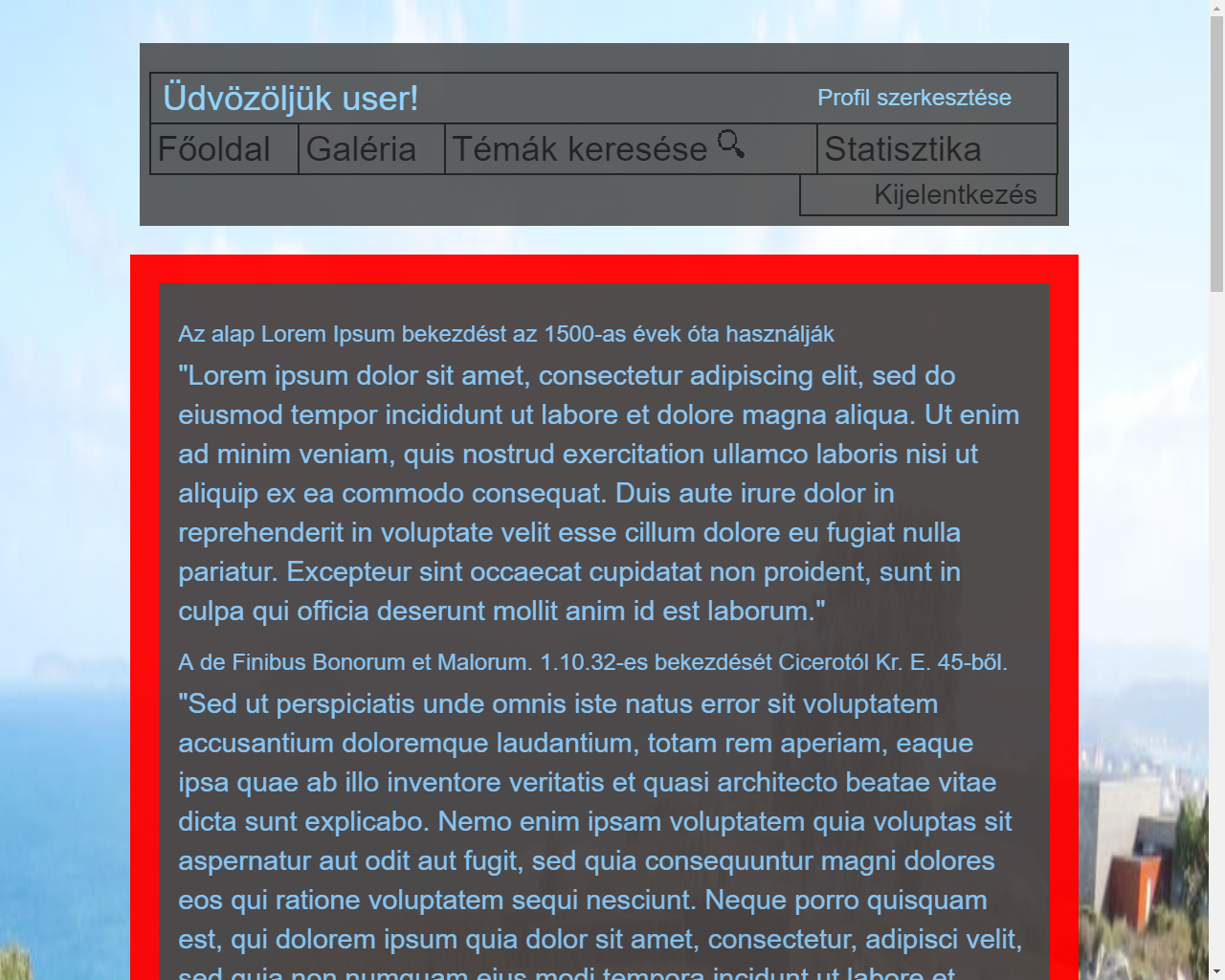
*képernyőkép: index.php (teljes felbontás)*

Regisztráció lehetőség kiválasztása után megadhatjuk a kívánt felhasználónevet emailt, majd a jelszót, szükséges ennek megerősítése ezek formai követelményeit JavaScript kód ellenőrzi

Ahogy belépünk az oldalra főoldal címszó alatt egy példaszöveg látható, mely kicserélhető akármilyen egyszerű HTML elemek halmazára

Ha a galéria navigációs menüpontra kattintunk, a szerveren tárolt képeket láthatjuk (jelenleg egybeszervezve) melyekre klikkelést követően azok nagyobb felbontásban új ablakban nyílnak meg

Ha a felhasználó bejelentkezett felül az *üdvözöljük* felirat után a felhasználóneve látható, mely barátságos, udvarias és képet ad és emellett nem rejt semmiféle biztonsági kockázatot



*képernyőkép: főoldal bejelentkezés után*

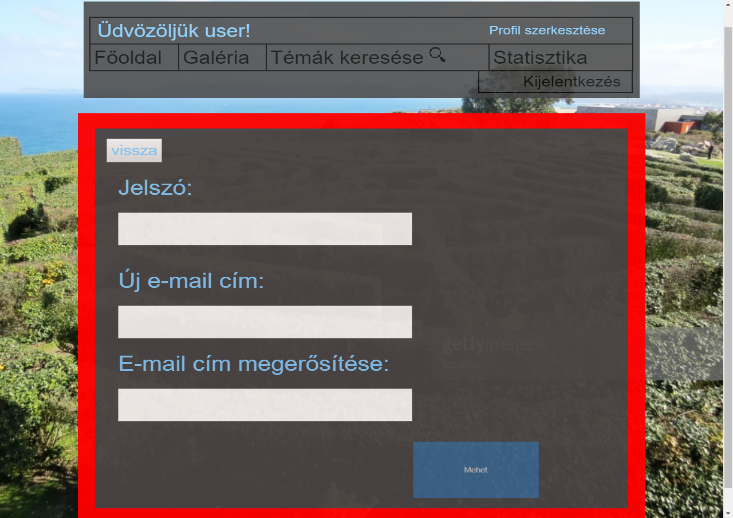
Bejelentkezés után lehetőség van még a profil szerkesztése gombra való klikkelés után a jelszó és az e-mail cím megváltoztatására (korábbi) jelszó beírásával a jogosultság ellenőrzése mellett

Illetve bejelentkezés után kijelentkezésre is van lehetőség

Funkció bemutatása: E-mail cím módosítása

1. Bejelentkezést követően a felhasználó, ha a profil szerkesztésére kattint lehetősége van az email cím vagy jelszó megváltoztatására

*képernyőkép a profil szerkesztésről*

1. A lehetőségek közül, ha a bejelentkezett felhasználó kiválasztja az „E-mail cím megváltoztatása” lehetőséget az új megjelent űrlap három főbb részre osztható

*képernyőkép e-mail cím megváltoztatásáról*

Vissza gomb Mellyel a fiók tulajdonosa visszatérhet a profil szerkesztési opciókhoz (Navigációs menüpont ugyanúgy működik, ha másfele szeretne navigálni)

Jelszó ellenőrzése illetve új e-mail megadása és megerősítése

Illetve a Mehet gomb, amely megnyomására, ha az adatok szintaktikailag megfelelők hibás adatok esetén tájékoztatja a felhasználót, és nem tölti fel az adatokat, míg megfelelő adatok bevitele után nyugtázza a sikerességet, majd tovább irányítja a felhasználót a megfelelő oldalra

Szintaktikai ellenőrzés:

A jelszónál a hosszúságot nem szükséges ellenőrizni, mivel az csak megfelelő hosszúsággal kerülhetett fel az adatbázisba illetve a beleírt adatot követő feltétel megakadályozza az SQLInjection-nél gyakran használt UNION parancs használatát, amellyel több táblát lenne lehetséges megjeleníteni – bővebben a biztonságról 🡪🡪🡪

Jelszó beírásakor ahogy az elvárható a beírt karakterek helyett pontok jelennek meg

Új e-mail cím:

A cím tartalmazzon ’@’ és ’-’ jelet, ebben a sorrendben, mindkettőből pontosan egyet-egyet

A ’@’ előtt a ’@’ és ’.’ között illetve ’.’ után is karakterek megléte szükséges

A karakterek mind az ABC betűi és nem írásjelek

Ha az új e-mail cím nem érvényes a beviteli lehetőségek alatt erre piros betűvel figyelmeztetés jelenik meg (amira az ellenőrzés valós időben történik).

Ha érvényes az új e-mail cím egyeztetés történik az új e-mail és az új e-mail megerősítése között, mely ha nem egyezik, szintén piros betűs figyelmeztetés által értesülünk róla, ha pedig egyezik végül a Mehet gomb kattinthatóvá válik

Ellenőrzés az adatbázisban:

Ha az elküldött e-mail cím és annak megerősítése „nyelvtanilag” megfelelő

…és a fiók tulajdonosa vagy bárki a számítógépnél a mehet gombra kattintott az oldal egy ajaxos post kérés segítségével csatlakozik az adatbázishoz, ellenőrzi, hogy jött-e felhasználónév (bejelentkezett felhasználó) e-mail cím és jelszó.

Ha ezek mindegyike érkezett a kérést megvalósító php összehasonlítja a felhasználónevet és jelszót az adatbázisban található felhasználói név-jelszó párosokkal, és ha talál ilyet, az e-mail címet megváltoztatja annál a felhasználónál, ha nem, nem teszi, küld egy visszajelzést sikerült-e, mely információt a korábban említett Mehet gomb funkciója feldolgozó

A felhasználó: Tipikus futtatási példák

Asztali:

1. Felhasználó elindítja a programot
2. Játék indítására kattint, hogy kipróbálja milyen is lehet
3. Új játékot választja, hogy elölről kezdhesse, tiszta lappal
4. Kiválasztja a számára megfelelő beállításokat (ha esetleg mégsem megfelelők később visszatér ide)
5. Játszik

Web:

1. A felhasználó az index.php oldalra megérkezik
2. Regisztrál
3. Bejelentkezik
4. Megnyitja a galériát, majd a számára érdekes képeket nagy felbontásban tanulmányozza

Biztonsági kérdések:

Adatbázis:

Hashelés: A jelszavak titkosítására az általam eddig tanult egyetlen módszer az md5 amelyet azonban túlságosan is könnyű kijátszani

Mit tudok javasolni?

Ha valaki jelszavakat szeretne tárolni a legbiztonságosabb szinte 100%-ban, ha azokat nem önmaga kezeli pl. facebook bejelentkezéssel lehetne összekapcsolni, egészen egyszerű, ha az ember komolyan gondolja

### Mit tehetek, ha mégis én magam szeretném a jelszavakat tárolni?

A jelszavak tárolása és titkosítása egy igen érzékeny tudomány, amely lényegében hétről hétre változik, ezért fontos, hogy a fejlesztő, ha ilyesmire adná a fejét nagyon friss, maximum egy-két hónapos bemutatókat keressen, melyekből tanulhat, máskülönben rendszere nem lesz naprakész.

Fontos, sőt törvény írja elő, hogy jelszó nem tárolható titkosítás nélkül online programban

Az sem megoldás, ha az adatokat egy kód kulcs segítségével titkosítjuk, majd mikor szükség van rá visszafejtjük olvasható formában, úgy dolgozunk vele, hiszen ha valaki hozzáfér az adatokhoz, könnyedén hozzáférhet a kulcshoz is!

Az Adobe nevű nagy cég is elkövette ezt a hibát 2013-ban, amikor felhasználók milliónak szivárogtak ki privát adatai, a kulcs nélkül szerencsére, viszont mivel a jelszó emlékeztetőket is megszerezték, sokak jelszavát fejtették meg a hozzátartozó felhasználónevek birtokában.

Hashalés A népszerű hashek nagyon könnyen visszafejthetőek, még a google szerver is egyszerűen visszafejti, csak meg kell kérni rá. pl. MD5 vagy SHA1

### Mi lehet akkor a legjobb saját megoldás jelszavak titkosítására?

Hashing and salting (Hashelés és sózás)

A só egy hosszú véletlenszerű stringet jelent mely minden egyes felhasználónál különböző

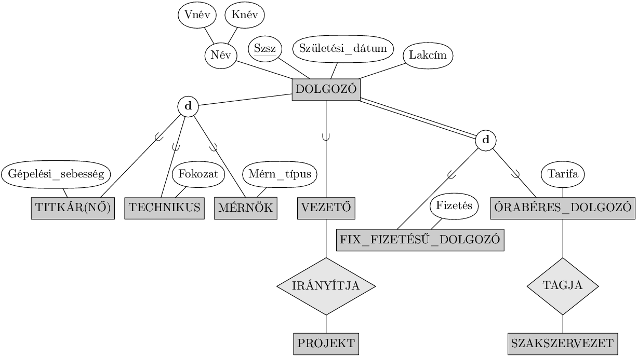
A hashnek elég sajátosnak kell lennie és megfelelnie a hashelési szabványoknak (például : Ha egyetlen byte megváltozik, nem számít hol, az egész hashelt eredmény gyökeresen változon meg, nézzen ki egész máshogy)

A módszer a következő: a jelszó hash-elése után tovább módosítjuk a tárolandó kódolt változót a „só”-val, így hiába egyezik két felhasználó jelszava a kódolt verzió egész más lesz és mégis kezelhető

Így megfelelő hosszúságú sóval és bonyolultságú titkosító módszerrel évszázadokig tarthat a jelszavak visszafejtése, ha a támadó nincs a só és a hash algoritmus birtokában, a titkosított jelszavakból nem lesz tömeges kiszivárgás

Adatbázistervezés:

Legnépszerűbb tervezési forma a relációs adatbázis séma, melynek lényege az egyes elemek közötti kapcsolat, ahol lényegében maga a kapcsolat a tábla amely tartalmazza az elemek összekapcsolását, amely elemek tulajdonságait külön táblákban tároljuk

Példa:

*kép:* [*http://meskobalazs.progmat.hu/tananyagok/fejezetek\_az\_adatbazisrendszerek\_elmeletebol/book.html*](http://meskobalazs.progmat.hu/tananyagok/fejezetek_az_adatbazisrendszerek_elmeletebol/book.html)

Minden fontos adatot tartalmazzon, többiből előállítható információt tárolni fölösleges, így annak elkerülés kiemelt prioritás (redundancia kerülése)

Irreleváns adatot ne tároljunk

Ne vesszen el információ

Lehető legkevesebb módosítás

Fontos a jellemzők közötti összefüggések megértése

### Ergonómia:

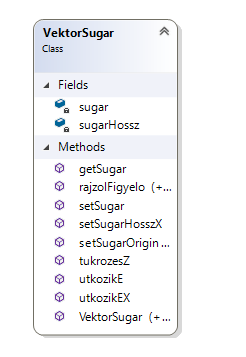
Az ember-gép munkakapcsolatával foglalkozó tudományág

Ismerős minden funkció, külön betanulásra nincs szükség

Gépekről minden szükséges információ elérhető

### Néhány szó az objektum orientált programozásról (OOP) az Alkalmazásomban:

Az asztali programozás során „Unity” ingyenesen letölthető 3D programkészítő rendszert használtam magának a játéknak a projektjéhez, mely gyökereitől eredve egészen a felszínig végig OOP módszert használ, ahogy a rendszer ablakainál én is, illetve magán a játékon belül is a kamerafigyelő (VektorSugar) is ezzel a módszerrel működik



Tesztelés:

A teszt egy olyan programrészlet, amely a szoftverfejlesztés bármely fázisában helyes eredményt ad, ha a program rendeltetésének megfelelően működik

Ugyanakkor hiába fut le minden teszt helyesen attól még nem biztos, hogy a program maga helyes

A programtesztelés négy szintje:

* Komponens teszt (egység és modell teszt)
* Integrációs teszt
* Rendszerteszt
* Átvételi teszt

### Tesztelés a programomban:

Az én alkalmazásomban két metódusra készült teszt, ebből az egyik csoport egyszerűen a metódus működtetését egy függvénybe csomagolja mely siker esetén igaz, egyébként hamis értékkel tér vissza

### Teszt 1:

A bejelentkezesAblakTest a „vizsgalHosszMegfE” nevű függvényt teszteli, melyet a program az online bejelentkezés vizsgálata előtt használ, hogy kiküszöbölje a nyelvtanilag nem megfelelő jelszó vagy felhasználónév páros szükségtelen vizsgálatát, ha azok bármelyike túlságosan rövid

* Három teszt: elég hosszú 🡪 true
* üres 🡪 false
* nem elég hosszú 🡪 false

Az első teszt a helyes bevitel („Labirintus”) esetén ellenőrzi a működést a következő képpen:

Megadtuk a file nevét, és a teszt mappából visszatérünk a fő program „bin/debug” mappájába ahol megkeressük a file nevet az alapértelmezett „exe” kiterjesztéssel.

Ezután eltároljuk, hogy sikeresen megnyílt-e a program, ha igen be is zárjuk, ezután sikeres a teszt, ha pedig nem a teszt egyszerűen sikertelen.

### Teszt 2:

A másik teszt a JatekValasztasAblak osztálynak a jatekfuttatas metódusát ellenőrzi, mely elindítja az új játékot az „új játék” gombra történő kattintás után

A kettes teszt az előzőhez hasonlóan működik, lényegében magát a tesztet ellenőrzi, hogy rossz adatok esetén miként működik a függvény.

Ha úgy ahogy elvárható a teszt sikeres.

Komplex alkalmazás: Összegzés:

### Eredeti tervek:

Egy teljesen jól működő alkalmazás mely később online játékmódra is alkalmas lesz

Magyar nyelvet támogatja

Offline játék folytatása mentések alapján

Agresszión hangsúly nincs a játék során

Ellenőrzött bejelentkezés

Villódzó fények kerülése

Hangulatos megfelelő licenszel rendelkező zenék használata

Megvásárolható, ám ingyenes legyen, lehetőség a készítő/készítők támogatására

3D motor nélkül készüljön (saját fejlődés a területen)

### Web:

Bejelentkezés, profil szerkesztés, kiírás főoldalra és topicosan elrendezett komment rendszer megvalósítása

### Adatbázis:

Megfelelő mennyiségű tartalom és információ

Végleges verzió:

### Asztali:

A játék még nem készült el, viszont a főbb grafikai komponensek a helyükre kerültek, ezenfelül érzékelhető miként fog működni a játék

Teljes mértékben magyar nyelvű

Irányítás még nem lehetséges pusztán egérmutató használatával, wsad-vel, iránygombokkal igen

A játék kreatív, a pálya elemeinek és akadályok leküzdésének módjával elégedett vagyok (vagy legalábbis a már megtervezett elméletével mindenképp)

A mentési lehetőség határidőre nem készült el, ám egy pálya így is játszható marad

Villódzás nem jellemző (kamera bugtól eltekintve)

A bejelentkezés ellenőrzött

Hangok még nem készültek hozzá

Internetes oldalon patreon által a projekt támogatható

A játék Unity 3D Motor segítségével készült az időre való tekintettel, melyre váltás jó tapasztalat és nagyban növelte a hatékonyságot

### Web:

Sikeres bejelentkezés, jelszó és e-mail változtatás, fejlesztők által felületre írás lehetséges

Topicos megoldás nem valósult meg

### Adatbázis:

Rendelkezik a szükséges tartalommal

Használatra megfelelő, a jelszavak jelenlegi biztonságától eltekintve

Fejlesztési lehetőségek:

A befejezettlen modulok elkészítése pl.

* A játék befejezése
* Web témák szerint rendezett fórum
* Jelszavak védelme, témák tárolása

Lehetőséget teremteni, hogy a játékosok egymás számára készíthessenek pályákat (erre szeretnék különösen nagy hangsúlyt fektetni)

Hangok hozzáadása a játékhoz

Online-mód

Statisztikák kiírása a felhasználók számára

Többplatformossá tenni az irányítást

Csak egeres irányítás játékban nem annyira gyakorlott felhasználók számára

Több pálya

Többnyelvűség

Beállításokban hang és fényerő szabályozása

Fontosabb akadályok a fejlesztés során:

Csak párat emelnék ki:

* Engine használata – készítése
* bonyolult 3D számítások
* viszonylag egyszerű problémák: pl. egy változóból másolatot szeretnék készíteni és nem az eddigit magát továbbvinni (shalow copy – deep clone)
* Összetett mátrixos szorzások használata
* Kamerakezelés
  + Saját osztállyal vizsgálat előre-hátra, de az a játékossal nem relatív
  + 90°-nál nagyobb fordulás esetén az esemény nem mindig következik be
* Összetett alakzatokhoz fizikai tulajdonságok rendelése – kocka
* Füst szimulálása – részecske rendszer
* Unity és blender nem teljes összhangja
  + animációk, részecske – rendszerek esetén
  + anyagok nem megfelelően érkeznek
  + ezek kezelése

Fejlődéseim:

Megtanultam, hogy három dimenzióban nem olyan egyszerű gondolkozni, úgymint: ami távolabb van annak nemcsak a vetülete kisebb, de ezentúl kevésbé is változik

Deep clone és shallow copy fogalmai és használata

Gyakorlat összetett osztályok használatával

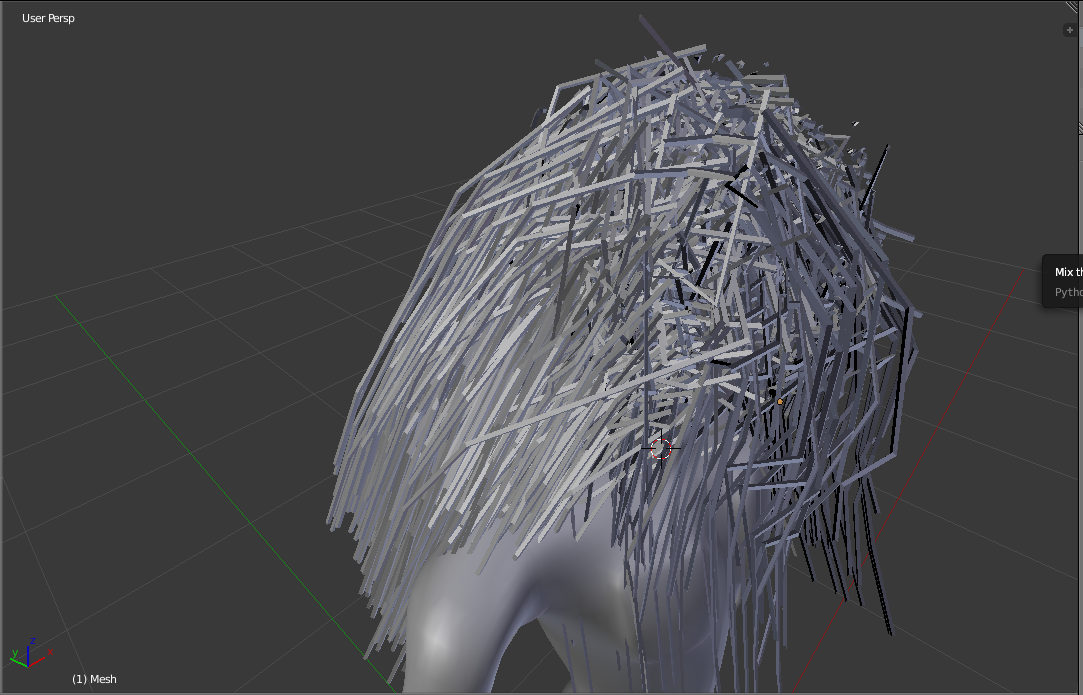
Elforgatás és transzformáció mátrixos forgatás segítségével – felszínesen

Unity kezelése

* felület használata
* Translate() és GameObjekt.transform.positon módosítása közötti különbségek
* transform.localposition
* euler angles set
* Camera raycast és annak ütközése objektumokkal
* UI kezelése

Amiről még sokat tanultam az az, hogy blenderben hogyan kell egy alakzatot lényegében bárhogyan formálni

* kijelölés akadályig
* vektorok pontjainak egyesítése
* görbe vonalak hatása
* mozgatás bármit g-vel
* elhelyezés



* forgatások
* szőr és haj szimulálása, materializálása
* élek között híd készítése
* stb…

# **Köszönetnyilvánítás**

Szeretném megköszönni tanáraimnak kihasználván ezt lehetőséget, hogy elindítottak, támogattak segítettek és tanítottak a programozás rögös útjain.

Először is szeretném megemlíteni Szabó Levente tanárurat a Pécsi Tudományegyetem oszlopos oktatóját, aki az alapokat megismertette velem, és habár viszonyunk nem volt egészen felhőtlen mégis világosabbá vált általa miket is érhetek el.

Szeretném megköszönni még Gyuris Csaba tanárúrnak, osztályfőnökünknek, akinek szellemes jóhangulatú ám komoly órái alatt tovább bővíthettem az addigi valamicske tudást ténylegesen használható tudománnyá érlelve azt.

Köszönöm Bálint Róbert tanárúrnak, általa egyszerű módokon szépen lassan közelebb és közelebb kerültem a weboldalak programozáshoz, egészen megszeretve azokat, meglátván azokban nemcsak a frontend, hanem a backend lehetőségeit.

Köszönöm még Péter Miklós középiskolai oktatómnak, hogy bevezetett a logikai gondolkodás világába a problémák egyszerűbb egységekre bontásának csodálatos természetét megismertetve. Köszönöm még korábbi tanáraimnak, akik szintén elősegítették ezt a folyamatot!

Szakdolgozat írásához felhasznált források:

*http://www.sze.hu/~heckenas/okt/architec*

*https://en.wikipedia.org/wiki/Software\_architecture*

*https://inf.mit.bme.hu/sites/default/files/edu/doktori/szvv/referatum2011/ISO9126\_(Csuka\_Zsolt).pdf*

*https://en.wikipedia.org/wiki/ISO/IEC\_9126*

*https://www.youtube.com/watch?v=8ZtInClXe1Q*

*https://hu.wikipedia.org/wiki/Adatb%C3%A1zis-tervez%C3%A9s*

# Plágium - Nyilatkozat

/Nyilatkozat szakdolgozat készítésére vonatkozó szabályok betartásáról/

Alulírott ………………….……………………… (OM azonosító :………….….………) jelen nyilatkozat aláírásával kijelentem, hogy a

…………………………………………………………………………………………………………………………………………………………………………………………

című szakdolgozat (a továbbiakban: dolgozat) önálló munkám, a dolgozat készítése során betartottam a szerzői jogról szóló 1999. évi LXXVI. tv. rendelkezéseit, valamint az intézmény által előírt, a dolgozat készítésére vonatkozó szabályokat.

Jelen nyilatkozat aláírásával tudomásul veszem, hogy amennyiben bizonyítható, hogy a dolgozatot nem magam készítettem vagy a dolgozattal kapcsolatban szerzői jogsértés ténye merül fel, a Szegedi Szakképzési Centrum megtagadja a dolgozat befogadását.

Tudomással bírok arról is, hogy a szerzői jog megsértése miatt az arra jogosultak velem szemben felléphetnek, s az egyéb jogkövetkezményekkel – így polgári jogi, szabálysértési jogi, illetve büntetőjogi következményekkel – járhat.

Szeged, 2019.03.31

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

tanuló aláírása