UNIVERSITATEA SAPIENTIA DIN CLUJ-NAPOCA
FACULTATEA DE ȘTIINȚE TEHNICE ȘI UMANISTE, TÎRGU-MUREȘ
PROGRAMUL DE STUDII CALCULATOARE

Sapi3D tour – UI

PROIECT DE DIPLOMĂ

Coordonator ştiinţific:
Ș.l.dr.ing. Szántó Zoltán

Absolvent:
Nagy-Serbán Tünde

2021

UNIVERSITATEA "SAPIENTIA" din CLUJ-NAPOCA **Viza facultății:**
Facultatea de Științe Tehnice și Umaniste din Târgu Mureș
Specializarea: **Calculatoarea**

LUCRARE DE DIPLOMĂ

Coordonator științific:	Candidat: **Nagy-Serbán Tünde**
Ș.l. dr. ing. Szántó Zoltán	Anul absolvirii: **2021**

a) Tema lucrării de licență:

b) Problemele principale tratate:

c) Desene obligatorii:

d) Softuri obligatorii:

e) Bibliografia recomandată:

f) Termene obligatorii de consultații: săptămânal

g) Locul și durata practicii: Universitatea Sapientia,
Facultatea de Științe Tehnice și Umaniste din Târgu Mureș

Primit tema la data de:
Termen de predare:

Semnătura Director Departament **Semnătura coordonatorului**

Semnătura responsabilului programului de studiu **Semnătura candidatului**

Declaraţie

Subsemnata Nagy-Serbán Tünde, absolvent al specializării Calculatoarea, promoţia 2021 cunoscând prevederile Legii Educaţiei Naţionale 1/2011 şi a Codului de etică şi deontologie profesională a Universităţii Sapientia cu privire la furt intelectual declar pe propria răspundere că prezenta lucrare de licenţă/proiect de diplomă/disertaţie se bazează pe activitatea personală, cercetarea/proiectarea este efectuată de mine, informaţiile şi datele preluate din literatura de specialitate sunt citate în mod corespunzător.

Târgu Mureș,

Data:

Extras

Extract

Cuvinte cheie:

SAPIENTIA ERDÉLYI MAGYAR TUDOMÁNYEGYETEM
MAROSVÁSÁRHELYI KAR
SZÁMÍTÁSTECHNIKA SZAK

Sapi3D tour - UI

DIPLOMADOLGOZAT

Témavezető:

Dr. Szántó Zoltán
adjunktus

Végzős hallgató:

Nagy-Serbán Tünde

2021

Kivonat

A dolgozat egy webalkalmazást mutat be, amely a Sapientia Erdélyi Magyar Tudományegyetem Marosvásárhely-i karának a 3D virtuális túráját foglalja magába.

A mai világban az emberek napjait nagyban befolyásolja a digitalizáció. A legtöbb embernél található legalább egy okostelefon, számítógép, laptop amelyek mellé társul az internet is így a lehetőségek száma végtelen.

A digitalizáció az emberek számára nagyon sok jó dolgot vezet be. Rengeteg problémát tudunk megoldani az internet és a digitális eszközök segítségével, mint például: számlák fizetése, távoli rokonokkal könnyebb a kapcsolattartás és nem utolsó sorban a virtuális túrák segítségével eltudunk jutni olyan helyekre ahová nem biztos, hogy az életben lesz lehetőségünk.

A dolgozatban egy webes applikációról van szó amley felhasznál egy 3D modellt az egyetemről létrehozva az egyetem úgy nevezett virtuális túráját. A túrán bárki résztvehet ezáltal betekintést nyerhet az egyetem fala mögé. Ezen kívűl informatív jelleggel is rendelkezik, mivel az alkalmazás számos információt megjelenít az egyetemmel kapcsolatban, mint például: különböző események (Sapi-Line-Tracer), szakkoordinátorok nevei, elérhetőségei.

Az alkalmazás ugyanakkor rendelkezik a "Vigyél el!" funkcionalitással, amely a felhasználó által kiválasztott helységhez mutatja meg az odavezető utat, sok segítséget nyújtva az egyetemre látogatoknak.Ezen kívül arra is van lehetőség, hogy a felhasználók saját maguk lépegessenek az egyetem modelljén így még jobban körbe tudják járni azt.

A rendszer webes felelületre készült és ennek köszönhetően majdnem minden eszközön meglehet tekinteni, úgy a számítógépen mint a laptopon és nem utolsó sorban a telefonon is. Az alkalmazás sok segítséget nyújthat az újjonnan érkező egyetemistáknak, vendég diákoknak és a vendég tanároknak is az egyetem fő épületében való eligazodásnál.

Kulcsszavak: webalkalmazás, 3D modell, virtuális túra.

Abstract

Abstract

Keywords:

Tartalomjegyzék

Ábrák jegyzéke

Táblázatok jegyzéke

1. fejezet

Bevezető

A mai gyorsan fejlődő világunkban a digitális eszközök a mindennapok elengedhetetlen részei. Nem sok olyan háztartás van ahol nincs egyáltalán legalább egy telefon, számítógép, laptop, okos tv. Az elmúlt években a digitális világ annyira fejlett lett, hogy lassan már ki sem kell mozdulnunk a házból és mindent eltudunk végezni. Például ki tudjuk fizetni a számláinkat, be tudunk vásárolni. Ezekből is következtethetünk, arra hogy egy jó internet kapcsolattal és egy közepes teljesítményű számítógép mellett már egy életet is letudunk élni.

Az elmúlt két évtizedben a „virtuális múzeum" [1] fogalmának meghatározása a gyors technológiai fejlődés következtében megváltozott. A mai rendelkezésre álló 3D technológiák a virtuális múzeumok esetén már nem csupán a gyűjtemények bemutatása az interneten vagy egy kiállítás virtuális bemutatója panorámás fotózás segítségével. Egy virtuális múzeum nem csak az oda látogatóknak hasznos, hanem már tanórákon is feltudják használni a tanárok mivel nem mindig sikerül elvinni a diákokat egy adott városba így egy ilyen virtuális túra segítségével a tanár betudja mutatni az adott múzeum látványosságait. Sőt a gyerekek ezáltal otthon is eltudnak barangolni más országok, kontinensek múzeumaiba virtuálisan.

Világunk fejlődése próbálja biztosítani, hogy egyetlen ember se maradjon le azon helyekről ahová nem tud eljutni, így egyes múzeumokat, iskolákat, kastélyokat, látványosságokat is betudunk járni otthon a négy fal között a 3D virtuális túrák segítségével.

Több okot is fel lehet sorolni annak érdekében, hogy miért is használatosak ezek a 3D modellekkel megalkotott virtuális túrák. Első sorban az új diákok már otthonról be tudnak nézni az egyetem

falai mögé így amikor elérkeznek az új év kezdéséhez akkor otthonosabban érezhetik magukat. Ez azért történhet meg mert már fogják tudni, hogy mit hol találnak nem kell segítséget kérjenek. Egy másik ok, hogyha az adott egyetem rendelkezik egy ilyen típusú modellel akkor jobban felhívhatja a figyelmet a diákok számára. Ez által az egyetem népszerűsítése is megtörténik. Ez mellet a vendég kutató, tanár vagy akár hallgató is könnyeben eltud igazodni az egyetem főépületében.

A 3D jelentése három dimenzió. Ez egy modell, amely tulajdonképpen matematikailag ábrázol egy háromdimenziós objektumot, amely lehet épület, virág, vagy akár egy ember is. A 3D modelleket széles körben használják az orvostudományban, magasabb szintű gyakorlati és elméleti kompetenciák elérésében. A 3D modellekkel nem csak tárgyakat hanem emberi folyamatokat is le lehet szimulálni.

Sok esetben egy szimulált 3D modell segít egy adott problémát jobban átlátni. Ilyen példa lehet amikor az orvos tudományban [2] nagyon szépen letudják előre játszani a műtéteket így biztosabbak a dolgukba és a kockázati lehetőségeket is csökkenthetik. Egy másik példa lehet egy hajóforgalom szimulációs rendszer [2] amely figyelembe veszi a hajókat, vízfelületet és az időjárási viszonyokat. Ezzel a rendszerrel próbálták megmutatni az olajszennyezést [2] a tengerekben, óceánokban. Ezen modellek segítségével a világ látványosságai elérhetővé válhatnak azon emberek számára is, akik nem jutnak el az eredeti országba, városba, hogy megtudják tekinteni az adott látványosságot.

A számítógépes grafikát [3] sok terülten alkalmazzák mint például az interaktív médiatervezésben, 3D túrák készítésében és videojáték iparban. Az elmúlt néhány év során számos technológia jelent meg a 3D virtuális túrák megjelenítésére az interneten. A virtuális túrák a felhasználóknak biztosítják az életszerű 3D környezeteket.

A tour (virtuális túra) szó jelentése utazás, kirándulás. A mai világban túrázni a virtuális világban is lehet. Egy ilyen virtuális túra célja, hogy fejlettebb szimulációs technikák segítségével a nézők élethű 3D-s képet kapjanak a meglévő helyről. Egy híres példa a Second-Life [3] ahol a felhasználók közötti interakció avatárokon keresztül zajlik. Ezen alkalmazáson belül interaktív 3D tour-ok vannak leképezve.

A 3D és a tour (virtuális túra) szavak összetételéből jön ki a 3D tour (3D virtuális túra) amely azt tükrözi, hogy a virtuális világban tudunk megnézni adott épületeket, kilátásokat, látványosságokat. Habár ezen virtuális 3D modelleken alapuló világ még nem tökéletes de folyamatosan fejlődik. A dolgozatomon belül egy ilyen modellről lesz szó amely a Sapientia Erdélyi Magyar Tudományegye-

tem Marosvásárhely-i karának egy részét tartalmazza. A projekt két részre van bontva, amely két államvizsga dolgozatot eredményez a 3D modell és a felhasználói felület. Közös munka a modellel történő attrakciók megoldása. Ilyen attrakció lehet a modellben való mozgás, közlekedés. Ezen dolgozatban a felhasználói felületről lesz szó.

Egy ilyen alkalmazás esetében nem csak maga a modell jelenik meg, hanem rajta kívül számos fontos információ, elérhetőség is. Dolgozatom célja bemutatni a Sapientia Erdélyi Magyar Tudományegyetem 3D Virtual Tour alkalmazás megalkotását, implementálását és nem utolsó sorban a felhasznált technológiákat.

2. fejezet

Célkitűzések

A projekt célja, hogy a felhasználók távolról is megtudják nézni az egyetemet. Ez elsősorban az új felvételizőknek és az első éves egyetemistáknak lenne hasznos, mivel ez által már otthon neki foghatnak áttekinteni az egyetem különböző részeit mint például, hogy hol található egy adott tanszék, vagy hol található a dékáni hivatal, vagy hogy hol található a könyvtár. Ilyen alkalmazás hasznos lehet úgy a diákok mint a tanárok, szakkoordinátorok számára, hiszen rengeteg apró de gyakori kérdésben tud segítséget nyújtani. Ugyanakkor egy egyetemen belül sokszor megfordulnak kutatók, vendég tanárok, külföldi diákok (Erasmus) így számukra is hasznos lehet egy ilyen jellegű alkalmazás.

Fontos szempont, hogy a felület legyen elérhető majdnem minden okos eszközön, ezért határoztunk úgy hogy a projekt egy webalkalmazás legyen, mivel ezt úgyan úgy meglehet nézni telefonon, számítógépen, laptopon.

A cél megvalósításának az első lépése az volt, hogy legyen egy felület ahol a Sapientia Erdélyi Magyar Tudományegyetem Marosvásárhely-i karának a 3D modelljét megtudjuk jeleníteni. A 3D modell a főépületet ábrázolja.

A modell megjelenítése mellett az alkalmazás egyik legfőbb célja megvalósítani egy olyan funkciót, hogy "Vigyél el!". Ez a funkcionalitás azt jelenti, hogy az új egyetemista diák, vendég tanár, vendég hallgató vagy akár vendég kutató bejelöl egy adott helyiséget ahová szeretne eljutni az egyetem területén és az alkalmazás a modell segítségével megmutatja az oda vezető utat.

Az alkalmazásnak szüksége van egy olyan funkcionalítás kivitelezésére is, ahol a felhasználók információkat (tanszékek, szakok, események) tudhatnak meg az egyetemmel kapcsolatban.

Cél az is, hogy az egyetemről egy link gyűjtemény kerüljön be az alkalmazásba. Ezt úgy kell érteni, hogy az adott tanszékekről, szakokról egy bővebb leírást mutatni, úgy hogy a linkeken keresztül átkerülünk olyan oldalakra ahol megjelennek bővebb információk az adott dologról. Ezáltal a diákok jobban eltudják dönteni, hogy az a szak amelyet kiválasztanak mennyire lesz jó számukra.

A hiteles információk, linkek közzététele érdekében szükség van user menedzsmentre. Ez azt jelenti, hogy bejelentkező felhasználók fogják tudni, szerkeszteni, megadni esetleg törölni az egyetemmel kapcsolatos információkat.

A bejelentkezéshez szükség van egy regisztrációra is, viszont nem akárki tud beregisztrálni az alkalmazásba. Az alkalmazásba csak egy Admin-on keresztül lehet regisztrálni, majd bejelentkezni.

A bejelentkező személyek számára biztosítani szeretném, a felhasználói adatok biztonságos eltárolását és kezelését is.

Mivel az alkalmazás első éves diákok számára készül és ők még nem ismerik az egyetem keretein belül szervezendő eseményeket sem, így az is a célok közé tartozik, hogy egy esemény naptárral is bővüljön az alkalmazás. Így az új diákok tisztában lehetnek, hogy milyen események lesznek az egyetem területén.

Szeretnék, egy olyan részt is biztosítani minden felhasználó számára ahol a saját véleményét tudja kifejteni. Ezen véleményeket figyelembe véve szeretném kijavítani az észlelt hibákat.

3. fejezet

Szakirodalom áttekintése

A világhalón való keresés során sok tanulmányt találtam, amely leírja hogy egy virtuális túra egyetemeken, múzeumokban, különböző látványosságoknál mekkora befolyásoló képességgel rendelkezik. Megtudhattam, hogy sok egyetemnek van ilyen jellegű túrája más más típusban. Van amelyik egyetem a virtuális túrát képek sorozatában képzelte el, van amelyik panoráma képekben, van amelyik videókban és nem utolsó sorban jönnek azok akik 3D modellekkel valósították meg az egyetemük virtuális túráját. A következő alfejezetekben bemutatok pár hasonló témájú alkalmazást, megközelítést.

3.1. Telkom Egyetem 3D kampusza

Indonézia egyik legnagyobb magánegyeteme a *Telkom University*[1] [4] is a virtuális 3D túrát alkalmazza az új diákok oda vonzására. A túrák tartalmaznak videó és képsorozatokat plusz 3d alapú modelleket is. Tulajdonképpen egy 3D web alapú túrát dolgoztak ki a 3D Vista használatával.

Miközben fejlesztették, ezt a túrát kutatásokat végeztek, hogy mivel lenne jó elkészíteni, más egyetemek milyen technológiákat alkalmaztak hazai területeken. A következő eredmények születtek:

- R F Rahmat előadó a Universitas Sumatera Utara (USU) egyetemen és társai: Az USU egyetem épületeiről információ szolgáltatásokat kivitelezetek.
- Moloo előadó a Mauritius-i egyetemen és társai: virtuális túrát hoztak létre a WebGL és a

[1] https://tour.telkomuniversity.ac.id/

SketchUp segítségével.

- Fujita a Tokió-i egyetem tagja és társai: Mobile robotokat használtak virtuális túrák elkészítéséhez.

Szeliski R. aki a fotó technikákat mutatta be [4], szerint is a nagy felbontású fotók és a 3D modellek együttes használatával nagyobb érdeklődési kört lehet elérni, mint ha csak külön használnák őket. Az egyetem kiemelt kutatási stratégia terve volt, hogy ne csak a diákokat vonzzák magukhoz, hanem az IKT (Információs és kommunikációs technológia) technológiák fejlesztésében is érjenek el fejlődéseket. Ennek érdekében használtak 3D modelleket és nagy felbontású fotókat együttesen. A diákok számára nagy érdekességnek számított és ezzel elérték azt, hogy a diákok érdeklődését felkeltették.

3.2. Mauritiusi egyetem 3D túrája

A *mauritiusi*[2] egyetem [3] is megalkotta saját 3D virtuális túráját. A megalkotást WebGL segítségével történt. A WebGL (Web Graphics Library) könyvtár 3D-s valós idejű megjelenítést kínál. A WebGL az OpenGL-ből származik , és API-t biztosít a 3D grafikához. A WebGL nem teljesen stabil és néhány algoritmusa nem hatékonyan látja el a feladatait. Különböző böngészők memóriahasználata és és végrehajtási ideje eltérő amelyek mellékhatásokat idézhetnek elő a megírt alkalmazásban.

Az egyetem is szintén valós időbeni megjelenítést alkalmaz és a megfelelő működés érdekében tesztelték az alkalmazást teljesítmény szempontjából, amikor több felhasználó csatlakozik egyidejűleg. A teszt eredményiből az egyetem arra következtetett, hogy minél összetettebb az objektum és ha még textúrával is rendelkezik akkor a teljesítmény nagyon csökken mivel a modell így nagyon nagy erőforrást igényel.

3.3. Virtuális rendszerek az Old-Segeberg városházán

A Németország-i Old-Segeberg [1] város házának is készítettek ilyen virtuális túrát egy Windows alapú interaktív szoftvert és egy virtuális valóság alkalmazást HTC Vive rendszerekkel. Mindkét rend-

[2]https://www.middlesex.mu/about-mdx-mauritius/campus-virtual-tour

szert kipróbálták a látogatók. A visszajelzések alapján van jövője az ilyen jellegű alkalmazásoknak is. A virtuális múzeum csúcspontjai a műalkotások pontos ábrázolása.

A Windows alapú interaktív szoftvert egy asztali számítógép segítségével tekinthették meg a felhasználók. A lényege az volt, hogy az oda látogató nem lépett be teljesen a virtuális világba csak kívülről tekintett bele míg a HTC Vive applikáció segítségével már a felhasználó végig ment a város házán virtuálisan. HTC Vive applikáció több interaktivitást nyújt a felhasználóknak viszont mindkét kifejlesztett rendszernek megvannak az előnyei és a hátrányai is. Majdnem minden háznál található egy PC számítógép így a Windows alapú rendszert könnyebben népszerűbbé lehet tenni mint a HTC Vive-ot.

3.4. Kuba-i Nemzeti Művészeti iskola virtuális túrája

Az innovatív technológiák új lehetőségeket biztosítottak a kulturális helyszínekről szóló információk gyűjtésére, elemzésére és megosztására. E technológiák közül a gömbszerű képalkotás és a virtuális túrakörnyezet segítségével megalkották a Kuba-i Nemzeti Művészeti iskolát [5] is, pontosabban a Nemzeti Balettiskolát. A virtuális túrán megnézhetőek a tantermek, a fő kupola és az előadó termek. A túrát azért hozták létre mivel az iskola már nagyon régi és rossz állapotban van így ezzel megtudják mutatni az érdeklődőknek, hogy milyen is volt régen az épület.

Ez a virtuális túra kompatibilis számítógépekkel, táblagépekkel és egyéb mobil eszközökkel. A virtuális túrában be vannak építve pdf-ek, linkek amelyek az adott részről bővebb információkat szolgáltatnak a túrázóknak. Összesen 14 gömbpanorámát, 96 hotspotot (műveletek pl. előre lépés, vissza lépés) és 40 linket. Ezen eszközök segítségével biztosítanak a felhasználóknak kényelmes virtuális túrázási lehetőséget. Az képernyőn megjelenik egy menü rendszer is amellyel tudjuk irányítani a túrát.

Ezen tanulmány szerint ennek a túrának a célja az volt, hogy a Balettiskola fennmaradjon a következő nemzedékek számára is legalább virtuális közegbe.

3.5. Játéktechnika alkalmazása virtuális túrák esetén

A virtuális túrák információt nyújtanak multimédiás úton a felhasználóknak, azt a benyomást keltve hogy valós időben navigálnak különböző helyeken. Egy sikeres túra jelentése, hogy a felhasználó eltudja hinni, hogy ő habár virtuálisan is de járt abban az adott helyiségben. Ahhoz hogy ez a benyomás létre jöjön a modell pontos ábrázolást kell tartalmazzon az adott helyiségről. Ezen tanulmány szerint is az ilyen túrák felhasználhatóak létesítmények népszerűsítésére mivel egy interaktív élményt biztosítanak. Az ilyen jellegű túrákat össze lehet hasonlítani a számítógépes játékokkal is. Hiszen ezek a játékok annyiban különböznek, hogy nem valós időben és nem valós helyeken történnek. Persze vannak kivételek, ahol a játék egy egy része magába foglal egy valós helyszínt is.

Az Egyesült Királyság [6] számos egyeteme használ ilyen jellegű túrákat annak érdekében, hogy az emberek távolról is betekintést tudjanak nyerni az egyetemek világába. A kutatás szerint tizennégy brit egyetemet tekintve mindegyik weboldalán találtak állóképgalériát és videogalériát viszont meglepő számnak számított hogy 3600 interaktív túrát is találtak. Azért meglepő szám mivel ezek a túrák még nem igazán elterjedtek.

Ezen tanulmány felmérte az Egyesült Királyság egyetemei körében mennyire használatosak a virtuális túrák. Kiderült, hogy a virtuális egyetemi túrák interaktívabbá tették az egyetemek weboldalait és nagyobb fokú szolgáltatást biztosítottak a felhasználók számára hiszen nem csak képeket, videókat láttak az egyetemekről, hanem maga az egyetemet is. Az egyetemek ezen túrák megalkotásában teljes mértékben kihasználták a számítógépes játékokhoz használt grafikai eszközöket. Az modellek segítenek bemutatni az egyetemeket, plusz útvonalakat is tervezhetnek a modellben így már előre fogják tudni, hogy mit hol találnak a felhasználók. A jövőben valószínűleg majdnem minden egyetem fogja alkalmazni az ilyen jellegű túrákat.

4. fejezet

Technológiai áttekintés

A célkitűzésnek megfelelően próbáltam keresni megfelelő adatbázist, keretrendszereket. A következőkben egy pár ilyen jellegű technológiáról lesz szó.

4.1. Adatbázisok

Adatbázisnak nevezzük az azonos jellemzőjű és többnyire strukturált adatok összeségét. Az adatbázisokon belül több fajta műveletet tudunk elvégezni [7]: karbantartás, tárolás, lekérdezés, szerkesztés, módosítás és nem utolsó sorban az adatok törlése is egy lehetőség. Ezen funkciókat egy adatbázis kezelő rendszer segítségével tudjuk elvégezni. Azonban különbséget kell tennünk az SQL (Structured Query Language) és a NoSQL (Not only Structured Query Language) között.

4.1.1. SQL

Az elmúlt harminc évben a relációs adatbázis volt az alapértelmezett strukturált adatlekérdezési nyelv. Világunk fejlődése miatt egyre több és nagyobb információ adatok robbantak ki így az SQL alapú adatlekérdezés [8] elveszítette hatékonyságát ezzel maga elé állítva azt a kihívást, hogy a nagyobb adatbázisok kezelése jóval megnehezedett. Ebből kifolyólag lehet arra következtetni hogy az SQL alapú szerverek hajlamosak nagy mennyiségű memóriát foglalni, biztonsági kockázatokat és teljesítményproblémákat elkövetni.

4.1.2. NoSQL

A NoSQL adatbázisokat azért alkották meg, mivel az SQL adatbázisok merev struktúrával rendelkeznek aminek következtében egy adott táblába nem tudunk kihagyni egy oszlopnak a feltöltését sem. Ennek következtében vagy fiktív adatokat, vagy üres mezőket kell megadjunk azokban a mezőkben amelyeket nem akarunk használni. A NoSQl adatbázisok sokkal rugalmasabbak lettek, fő céljuk az adatok könnyű tárolása és visszakeresése függetlenül a szerkezetektől és tartalomtól. Automatikusan kezelik az adatkezelést, hibajavítást amelyek költségmegtakarítás szempontjából is fontosak [8].

4.1.3. SQL vs NoSQL

A relációs adatbázisok az egyszerűség miatt leggyakoribb adatbázistípusok. Az adatok több táblára vannak bontva amelyekhez egyszerűen hozzá lehet férni. A crud műveletek nagyon egyszerűen elvégezhetőek az SQL által megadott szintaxisok betartásával. Ilyen típusú adatbázis kezelő rendszerek a következőek: Oracle, SQL Server, MySQL, PostgreSQL stb. A folyamatos adatmennyiség miatt a relációs adatbázisok hátrányba kerültek a nem relációs adatbázisokkal szemben. A nem relációs adatbázisok sokkal gyorsabban és hatékonyabban tudják elvégezni feladataikat mint a relációsak. Nem relációs adatbázis típusú rendszerek a következőek lehetnek [9]: Firebase, MongoDB, GraphQL stb. Sok adat esetén, ha a hatékonyságra törekedünk akkor érdemesebb a nem relációs adatbázisokat használni.

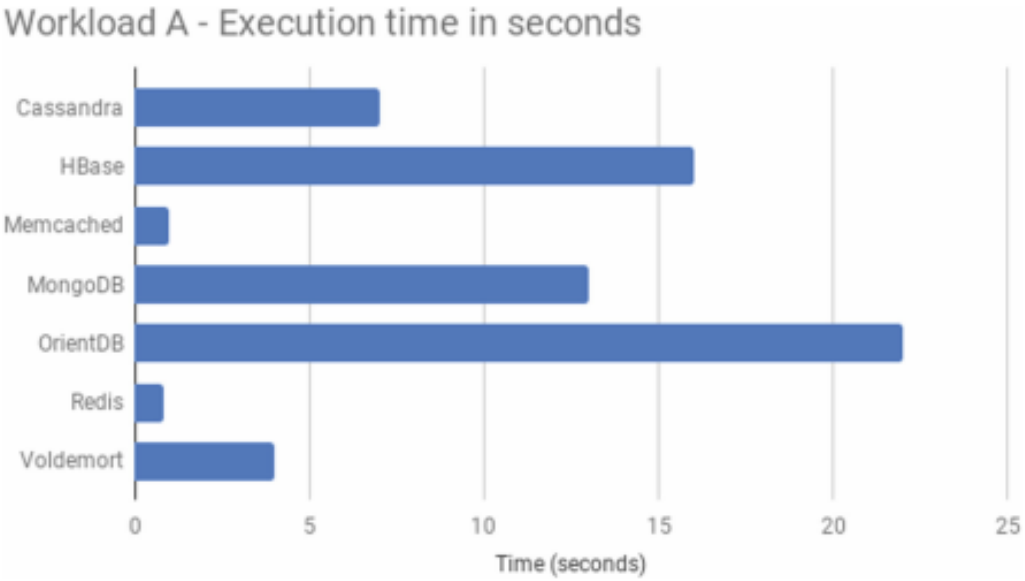
Több tanulmány is szól az adatbázisok teljesítményéről. A következőkben bemutatnék egyet. Ebben a tanulmányban három tesztet végeztek el. A kísérletben szerepelnek a következő NoSQL adatbázis kezelő rendszerek: Cassandra, HBase, Memcached, MongoDb, OrientDB, Redis és a Voldemort [10]. A tanulmányban három tesztet végeztek el. Mindhárom tesztben rendelkézsre állt 6.000.000 rekord. Ez a rekord szám nem is túl sok viszont nem is kevés. A tesztek az adatbázisok teljesítményét tesztelték olvasás, írás és frissítés esetén így elegendőnek bizonyult ennyi adat is. A teszteket időre alpazták, hogy a különböző adatbázisok hány szekundum alatt végzik el a különböző teszteseteket.

Az első tesztben a munkaterhelés fele olvasást és fele frissítést tartalmazott. Az eredmény a 4.1 ábrán látható. Az ábrát tekíntve láthatjuk, hogy a legjobb időt a Redis és egy kevéssel lemaradva a Memcached teljesítette. A legrosszabb teljesítményt az OrientDB mutatta.

A második tesztnél a munkaterhelés 5.000 véletlenszerű olvasás volt. Az eredmények 4.2 ábrán láthatóak. Az első teszthez hasonlóan itt is a Redis és a Memcached teljesített a legjobban viszont utolsó helyre már a HBase került.

A harmadik tesztben a munkaterhelés 5.000 véletlenszerű frissítés volt. Az eredmények 4.3 ábrán láthatóak. Az első helyeket ismét a Redis és a Memcached szerezte meg míg az utolsó helyre visszakerült az OrientDB.

A tanulmányban leírt teszteket és NoSQL adatbázisokat tekintve a legjobb választás a Redis vagy a Memcached lenne. A legrosszabb választás az OrientDB lenne. A tanulmányban résztvevő többi adatbázis teljesítménye körülbelül megegyező volt, közepes teljesítménnyel, így a tanulmányt tekintve ezeknek a használata is megfontolandó.



4.1. ábra. *6.000.000 rekord, 50% olvasás, 50% írás esetén*[1]

[1] https://www.researchgate.net/figure/Workload-A_fig1_332028074

4.2. ábra. *6.000.000 rekord és 5.000 véletlenszerű olvasás esetén*[2]

4.3. ábra. *6.000.000 rekord 5.000 véletlenszerű frissítés esetén*[3]

[2]https://www.researchgate.net/figure/Workload-C_fig2_332028074

[3]https://www.researchgate.net/figure/Workload-H_fig3_332028074

4.2. Webes keretrendszerek

A keretrendszerek [11] napjainkban felkapott rendszerek lettek a felhasználók között. Ezek a rendszerek sokoldalúak, robusztusak és hatékonyak. A különböző alkalmazások fejlesztéséhez az ilyen jellegű rendszerek segítségével csak a magasabb szintű funkcionalitások elvégzésére kell koncentrálni. Erre az a magyarázat, hogy a keretrendszer gondoskodik az alacsonyabb funkcionalitásokról, amelyek már rengeteg tesztelt kódot tartalmaznak, így mi ezeket a funkcionalitásokat nem kell külön megírjuk és leteszteljük, hogy helyes-e a megírt kód. Számos előnyt lehet felsorolni, hogy miért jó ha használjuk ezeket a rendszereket:

- Elősegíti a tervezési minták megfelelő kialakítását.
- Biztonságosabb kódolás.
- A redundáns kód elkerülése.
- Következetes kódfejlesztés kevesebb hibával.
- Megkönnyíti a kód tesztelését és a hibakeresést is.
- Az alkalmazás fejlesztéséhez szükséges idő lecsökken.

Az 4.4 ábrán látható egy rangsorolás a Github és Stack Overflow által készített pontozások alapján a Front-end-et megvalósító keretrendszereknek. A Front-end foglalkozik a kliens felületek megjelenítésével. Az ábrán látható hogy az első helyen a React van összesített 98 ponttal. A második helyen áll az ASP.NET MVC 95 ponttal viszont itt észlelhető az is, hogy a Githubról nem jött pontozás erről a keretrendszerről. Az ábrán látható további keretrendszerek úgyan annyi összpontozást kaptak. Az ábra alapján a legjobb döntés egy webes keretrendszer választásra a React lenne visszont a többi keretrendszer sincs sokkal lemaradva az első helyezettől.

Framework	Github Score	Stack Overflow Score	Overall Score
React	99	97	98
ASP.NET MVC		95	95
Angular	91	96	93
Ruby on Rails	87	99	93
AngularJS	90	97	93
Vue.js	100	87	93

4.4. ábra. *Webes keretrendszerek Github és Stack Overflow pontozások alapján*[4]

2021-ben a 10 leghasználatosabb Back-end keretrendszer a Spring - Spring Boot, NodeJS, Laravel, Django, Flask, Ruby, Play, Asp.Net, CakePHP és Symphony [12][13]. A Back-end segít a Front-end-nek egy dinamikus alkalmazás elkészítésében. Ez azt jelenti, hogy a Back-end biztosítja az adatokat a Front-end-nek.

A következőkben olvashatunk részletesebben néhány keretrendszerről mint például: az Angular, Vue.js, Spring Boot és a NodeJs.

4.2.1. Angular

A Google munkatársai 2008-ban fejlesztettek a JavaScript alapú Angular keretrendszert [14]. Abban az időben a webhelyek zöme többoldalas alkalmazás megközelítésén alapult. A több oldalas alkalmazások lassúnak bizonyultak így bevezették az egy oldalas alkalmazásokat. Az egy oldalas alkalmazások abban jobbak a több oldalas alkalmazásoknál, hogy weboldal betöltése folyamán nem kell mindig mindent újra betölteni, hanem csak azok a részek töltődnek be ahol változások fognak megjelenni. Az Angular volt az egy oldalas alkalmazások első kerete. Az egyik fő előnye, hogy a felhasználók egyszerű struktúrával kell dolgozzanak. Megtanulják az Angular sajátos felépítését ezáltal gyorsan és optimálisabban tudnak benne fejleszteni. Az sem elhanyagolható, hogy a fejlesztők egy részletes és egyértelmű dokumentációval szolgáltak a felhasználóknak.

[4]http://hotframeworks.com/

4.2.2. Vue.js

A Vue.js (röviden: Vue) [14] tekinthető az egyik legújabb keretrendszernek. Hasonlít az Angularhoz, mindkétt keretrendszer TypeScript típusú. Használható kisebb, egyszerűbb projekteknél. Mindig egy oldalas alkalmazásokat lehet benne elkészíteni ami a felhasználói élményt segíti elő. Fő érdeme a skálázhatóság és különlegessége, hogy egy nyílt forráskódú közösség fejlesztette ki, nem pedig egy nagyobb vállalat, így a problémák megoldására rengeteg segítséget lehet kapni. Komponens alapú keretrendszer amely azt jelenti, hogy komponenseket különböztetünk és jelenítünk meg. Egy komponensen belül tudunk írni HTML, CSS és Script elemeket is. A megjelenítés egy oldalon történik ezért szükséges használni a útválasztást (rout).

4.2.3. Angular vs Vue.js

Mindkét keretrendszernek megvannak az előnyei 4.1 táblázat és a hátrányai is 4.2 táblázat.

4.1. táblázat. *Előnyök Angular és Vue.js között [15]*

	Angular	**Vue.js**
1	TypeScript használata	TypeScript használata, részletes dokumentáció
2	Részletes dokumentációval rendelkezik	Egy oldalas alkalmazások készítése
3	Gyorsítja a fejlesztést	Könnyű integráció a meglévő struktúrákba
4		Kihasználja virtuális DOM előnyeit
5		Sebessége és rugalmassága optimális

4.2. táblázat. *Hátrányok Angular és Vue.js között [15]*

	Angular	**Vue.js**
1	Számos különféle struktúrát kínál, nehezíti a tanulást	Kevesebb erőforrást kínál
2	Lassabb teljesítmény mert működik a reális DOM	

4.2.4. NodeJs

A NodeJs [16] egy olyan szoftver platform, amely a Chrome V8 JavaScript futási idején épül fel. Fontos tulajdonsága, hogy skálázható ezért is sokan használják. Eseményvezérelt, nem blokkoló I/O modellt használ, amely könnyűvé és hatékonnyá teszi a valós idejű alkalmazások megvalósítását, amelyek elosztott rendszeren futnak át. Tulajdonképpen közvetlenül a natív gépi kódra fordítja le a JavaScript-et, amelyenek hatására az adatformátum egy lépése megszűnik, ezzel növelve az alkalmazás sebességét. Legtöbb esetben a NodeJs használatakor adatbázisnak NoSQL adatbázisokat választanak.

NodeJs a következő különlegességeket tartalmazza [17]:

- A NodeJS alkalmazások fejlesztésének elindítása könnyű.
- Az agilis fejlesztési módszertant követi
- Alkalmas a skálázható alkalmazásfejlesztési szolgáltatásokra.
- Nagy projekteknél gyorsabban működik mint a Java.
- Hatalmas könyvtárakkal rendelkezik, amelyek segítik a fejlesztőket.

4.2.5. Spring Boot

A Spring keretrendszer [18] egy programozási és konfigurációs modellt kínál a Java alapú alkalmazásokhoz. A Spring Boot [19] célja a Spring alkalmazás fejlesztés egyszerűsítése. Megtalálhatóak benne a következő tulajdonságok és különlegességek [17]:

- Automatikus konfigurációk - Az alkalmazások Springként való működése érdekében.
- Indítófüggőségek - Biztosítja a felhasználóknak a szükséges függőségek (dependency) beillesztését. Ilyen lehet a Maven, Hibernate validátor, adatbázis elérések stb.
- Parancssori tolmács.
- Működtetés - A console-ba megjelennek az alkalmazás működésével kapcsolatos információk. Ilyen információ lehet a hiba, az elvégzett művelet stb.

- Radikálisan gyorsabb és széles körben hozzáférhető, érthető Spring fejlesztést nyújt.
- Számos funkciót kínál: beágyazott szervereket, metrikákat, ellenőrzéseket, külső konfigurációkat.
- Egyszerű, minden eszköz és operációs rendszer támogatja.
- Beépített nyelvbiztonsági funkciókkal rendelkezik, amelyeket a Java Compiler beágyaz.
- Robusztus kódot alkalmaz.
- Integrációs képesség jó.
- Beágyazott HTTP-kiszolgálókat, például Jetty, Tomcat használ és egyszerűen teszteli a webes alkalmazásokat.

4.2.6. Spring Boot vs NodeJs

A Spring Boot és a NodeJs közötti néhány előnyt [20] az 4.3 táblázatban tudjuk megtekinteni.

4.3. táblázat. *Előnyök Spring Boot és NodeJS között*

	Spring Boot	**NodeJs**
1	Java nyelv már nagyon ismeretes így ha elakadás van könnyű segítséget kérni/keresni	A JavaScript futásidejének köszönhetően gyors az adatfeldolgozásban
2	Többszálú programozás	Memória takarékos
3	Nagyszámú erőforrás áll rendelkezésre (például: az autentikáció már előre megvan írva)	NPM(Node Package Manager)-nek köszönhetően egyre több szolgáltatás érhető el.

Minden keretrendszer között találhatunk előnyöket is meg hátrányokat is. A Spring Boot és a NodeJs között előnyöket az 4.3 táblázatban már tárgyaltuk. A következőkbe néhány hátrányról [20] lesz szó:

- A NodeJS nem tud hatékonyan teljesíteni nagy számítások esetén.

- A NodeJS mivel még új technika, ezért nincs teljesen kifelesztve, tesztelve így sok olyan hibába ütközhetünk amire nincs ideális megoldás.
- A Spring Boot legnagyobb hátránya, hogy a memóriaigénye nagyon nagy.
- A Spring Boot egy másik hátránya, hogy a hiba keresés meglehetősen nehéz mivel sok beépített kód található benne.

Összefoglalva mindkét technológia a maga módján megfelelő különböző alkalmazások tervezésénél. Ha az alkalmazás sok bemeneti/kimeneti feladatot (például: sok regisztráció) tartalmaz akkor mindenképp a NodeJs-t érdemes választani. Viszont ha biztonságos és önálló alkalmazást tervezünk amely intenzív CPU használatot igényel akkor a Spring Boottal érdemes foglalkozni.

5. fejezet

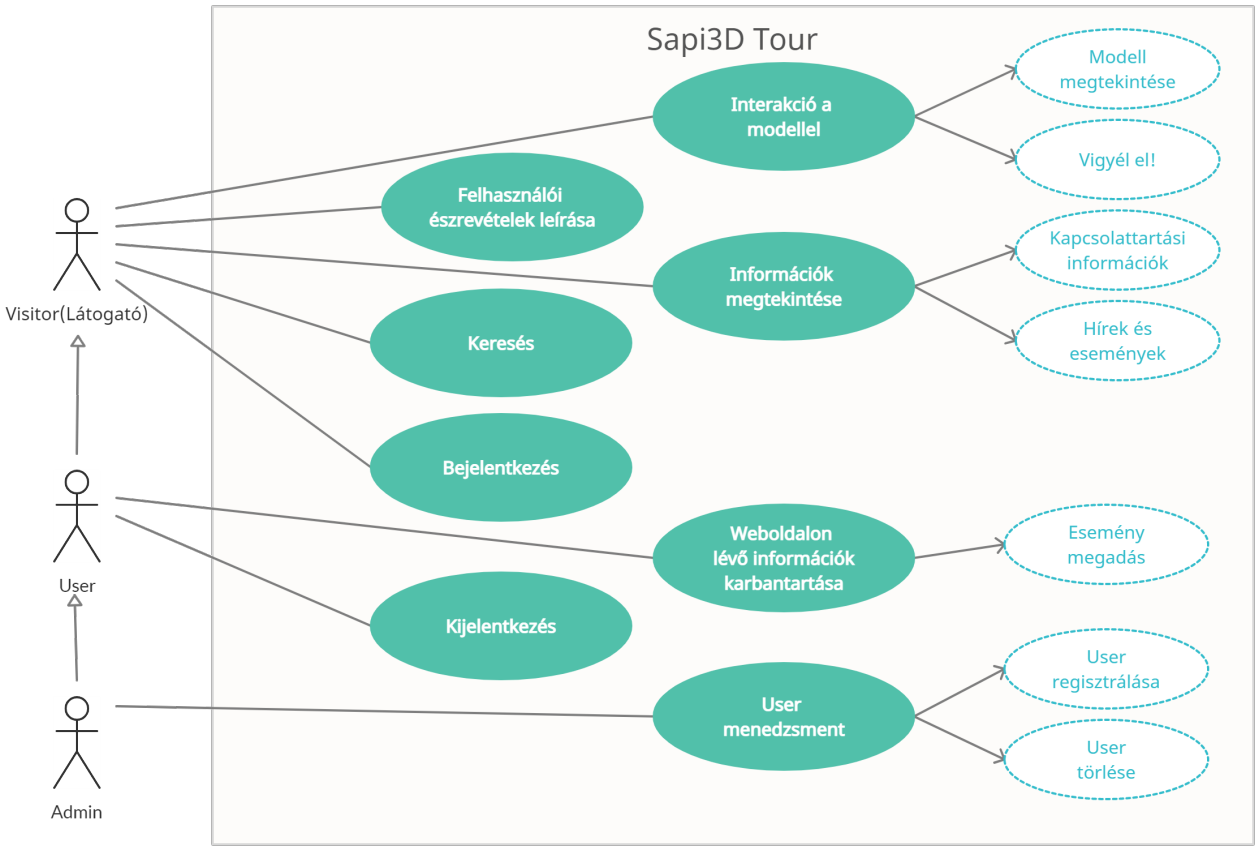
Követelmény specifikáció

5.1. Felhasználói követelmények

A Sapi3D alkalmazás web alapú, ezért mindenki számára elérhető. A fő célja, hogy egy 3D modellként jelenítse meg a Sapientia EMTE Marosvásárhely-i karának a fő épületét, illetve ennek fontosabb helyeit, mint pl. tanszékek, titkárság stb. A rendszer fontosabb funkcionalitásait és az ezeket igénybe vevő szerepköröket a 5.1 ábra szemlélteti.

Az alkalmazás eléréséhez, használatához egyetlen feltételnek kell eleget tenni, amely az internet kapcsolat megvalósítása lenne. A rendszer megértésének érdekében tekintsük meg a 5.1 ábrát amely bemutatja a rendszert, amit három különböző felhasználó vehet igénybe. A következő felhasználói szerepkörök vannak: Visitor, User és Admin, az utóbbi kettőhöz bejelentkezés szükséges.

Visitor (látogató), amelynek lehetősége van megtekinteni az elkészített weboldalt, igénybe veheti a "vigyél el" opciót és nem utolsó sorban saját kezűleg is végig tud menni az egyetem 3D modelljén. Vannak olyan Visitor-ok akik betudnak jelentkezni, így átalakulnak User-ré. A User az aki felelős az alkalmazás karbantartásáért. Az admin felhasználó az, akinek a rendszerben van a legnagyobb felelőssége. Ő felel azért, hogy mely visitorok kaphatnak engedélyt a bejelentkezéshez. Ez mellett az ő hatáskörébe tartozik, hogy ki lesz kitörölve a rendszerből. A karbantartás alatt kell érteni azt, hogy az oldalon megjelenő információk napra készek legyenek.



5.1. ábra. *A rendszer használati eset diagramja*

Bármely felhasználó, aki egy böngészőből megnyitja az oldalt, a Visitor kategóriába kerül. Ez a fajta felhasználó megtekintheti a 3D modellt, körbe sétálhatja és el tud jutni pl.titkárságra. Amint megnyílt az oldal rögtön látható az egyetemről készített modell. Ezen a modellen tud nézelődni, esetleg körbe is tudja járni, vagy adott helységekre el is tud jutni (például: titkárság, adott tanszék). Ezen kívül lehetősége van információk, elérhetőségek, események részleteinek elolvasására is. Minden Visitor ugyan akkor leírhatja saját véleményét, meglátásait az oldallal kapcsolatban is. A fent említett műveletek elvégzéséhez nem kell sem bejelentkezés, sem regisztráció.

Amennyiben a Visitor bejelentkezik átkerül a User kategóriába. A bejelentkezéshes szükséges megadni egy már regisztrált e-mail címet és egy már hitelesített jelszót is. A User engedélyezése nem regszitráció alapján történik, hanem az admin joggal rendelkezők osztják ki, mivel a rendszer úgy van megtervezve, hogy nincs direkt regisztráció, hanem csak egy Admin tudja beregisztrálni az

új User-eket. Egy User képes különböző műveletek elvégzésére, mint például: eseményeket megadni, az egyetemmel kapcsolatos információkat módosítani. Ezeken kívül a Usereknek lehetősége van a kijelentkezésre is.

A regisztrációs feladatot csak egy admin jogosultsággal rendelkező felhasználó végezheti el. Azért ezt a megoldást választottuk, mivel ez a weboldal az egyetemmel kapcsolatos információkat dolgozza fel, így ezekhez nem mindenki férhet hozzá. Nem csak a regisztrálás tartozik ehhez a feladatkörhöz, hanem a userek törlése is az ő feladata. Ezen plusz műveletek mellett az Admin is szintén elvégezheti a User és a Visitor műveleteit is.

5.2. Rendszerkövetelemények

5.2.1. Funkcionális követelmények

Visitor

- Interakció létesítése a 3D modellel. Ez azt jelenti, hogy a megjelenített modell fölött található gombok használata. A gombok bíztosítják az előre hátra jobbra és balra menést.
- A "Vigyél el" funkció használata. A modell fölött található legördülő listából való kiválasztás esetén egy új gomb megjelenésével és annak használatával a Bejárattól megmutatódik az út a kiválasztott helyre.
- Fontos egyetemi helyek menüpont megtekintése, ahol az adatbázsban található adatok jelenítődnek meg. A címkre kattintva megjelennek a bővebb információt tartalmazó linkek. A címek mellet megjelenik egy térkép ikon is, amelyre ha rákattintanak akkor vissza kerülnek a 3D modellhez és a "Vigyél el" funkciót használva ismét megmutatódik az út a bejárattól a kiválasztott helyig.

User

- Jelszó hitelesítése. Az e-mail-ben kapott validációs tokent felhasználva a jelszó hitelesítés oldalon megadni a jelszót, a megfelelő formátummal (Legyen benne legalább kisbetű,nagybetű és

szám. Legyen legalább 5 karakter hosszú).

- Bejelentkezés a regisztrált e-mail és hitelesített jelszó megadásával történik, amint a User megnyomja a "BEJELENTKEZÉS" gombot.
- Hozzáférés az adatbázisban tárolt adatokhoz.
- User képes szerkeszteni a saját adatait (név, e-mail cím, telefonszám), ha rákattint a jobb sarokban lévő ember ikonra. Ezek után egy ablakba előjönnek az adatai amelyek szerkeszthetőek. A szerkesztés után a "ADATOK SZERKESZTÉSE" gombra kattintva az adatok bekerülnek az adatbázisba. A megadott adatok elegett kell tegyenek a megfelelő formátumoknak.
- Az egyetemi részleg megadása, szerkesztése funkciónál egy rádiógomb segítségével ki tudja választani a User, hogy új részleget szeretne megadni, vagy már egy létező részleget szeretne módosítani. Ha a részleg megadását választja akkor megkell adja a részleg nevét (példaul: Dékáni Hivatal) és egy linket amely elvisz a részleget leíró oldalra (például: https://ms.sapientia.ro/hu/munkatarsak/d hivatal). Ezek után a "HOZZÁADÁS" gomb megnyomásával az adatok bekerülnek az adatbázisba. A szerkesztés esetén lehet módosítani a nevet is és a linket is. A "SZERKESZTÉS" gomb segítségével az adatbázisban szereplő adatok átíródnak.
- A szak megadása és szerkesztése funkciónál szintén egy rádiógomb segítségével tudja eldönteni, hogy megadni szeretne egy új szakot, vagy a meglévőket szeretné modósítani. Az új szak megadásánál a következő információkat kell megadni: szak neve; szakkoordinátor neve; szakkoordinátor e-mail címe; terem száma, ahol a szakkoordinátor tudja fogadni az érdeklődőket; egy lineket a szak részletes leírásához és egy legördülő lista segítségével, hogy melyik tanszékhez tartozik az adott szak. A "HOZZÁADÁS" gomb segítségével az adatok bekerülnek az adatbázisba. A szerkesztés esetén a fent említett adatokat lehet szerkeszteni. A szerkesztés akkor lesz végleges ha a User megnyomja a "SZERKESZTÉS" gombot.
- Egy részleghez, nem csak szakot lehet megadni hanem eseményeket, tevékenységeket is ha az Egyebek hozzáadását használja a User. Itt megkell adni egy legördülő lista segítségével, hogy a mely részleghez tartozik (például Villamosmérnöki Tanszék), egy nevet (például: Sapi-Line-

Tracer), és egy lineket ahol több információ van leírva az eseményről, tevékenységről. Itt is szintén a "HOZZÁADÁS" gomb megnyomásával kerülnek be az adatok az adatbázisba.

- A User ha elvégezte a hozzáadásos, szerkesztéses feladatait, akkor a jobb felső sarokban található "KIJELENTKEZŐ" gombra kattintva ki is tud jelentkezni.

Admin

- Hozzáférés az adatbázisban tárolt adatokhoz.
- Userek regisztrálása és az adatok validálása a teljes név, e-mail cím, telefonszám megadásával. Az e-mail cím és a telefonszám eleget kell tegyen a megfelelő formátumoknak. ezek után a "HOZZÁADÁS" gomb lenyomásával az adatok eltárolása a NoSql adatbázisba. Egy validációs token generálása és elküldése a Usernek a jelszó hitelesítéséhez.
- Userek törlése: a User e-mail címének kiválasztása egy legördülő listából, majd a "FELHASZNÁLÓ KERESÉSE" gomb lenyomásával a User adatainak megkeresése. Ezek után a "TÖRLÉS" gomb használatával a User adatainak kitörlése az adatbázisból.

5.2.2. Nem funkcionális követelmények

- A rendszer használatához a felhasználóknak szüksége van Internetre és egy digitális eszközre amelyen megtudják nyitni a webalkalamzást.
- A digitális eszköz lehet laptop, asztaligép, tablet vagy akár telefon is.
- A felhasználók nincsennek egy adott operációs rendszerhez kötve.
- Az operációs rendszer lehet Windows, Linux, Android és iOS is.
- Külön telefonos applikáció még nincs, viszont a telefonokon található böngészők bármelyikében meglehet tekinteni az alkalmazást.
- A regisztrált felhasználók adatai egy adatbázisban vannak eltárolva, amely nem lokálisan van jelen a felhasználók eszközén.

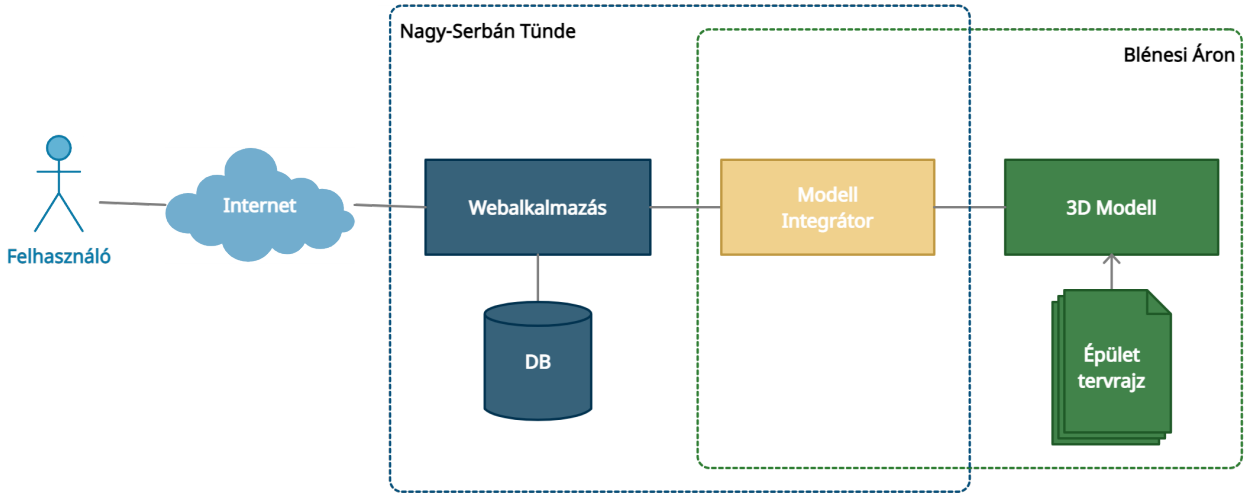
- Az azonosítás a Spring Boot beépített autetikációs moduljával történik meg.
- A bejelentkezéshez szükséges egy e-mail cím és egy hitelesített jelszó.
- A jelszó hitelesítéséhez a felhasználó kap egy e-mailt ahol tudja hitelesíteni jelszavát.
- Az e-mail tartalmaz egy tokent is amely segítségével tudja igazolni magát a felhasználó.
- Az e-mail lejárati idővel rendelkezik.

6. fejezet

Tervezés

6.1. Az rendszer architektúra

A rendszer több részből áll, melyen két diák is dolgozik, így két dolgozat is készül. Egyrészt a 3D modell, és másrészt az ehhez tartozó UI. Ezeket foglalja össze az 6.1 ábra.



6.1. ábra. *A teljes rendszer architektúrája*

A 3D modell tervezését, implementálását a kollégám, Blénesi Áron készíti el míg a webalkalmazást, adatbázis tervezést én. A két projektet egy komponensen keresztül kötjük össze. Ez a komponens a web alkalmazáson belül van létrehozva, és általa jelenítődik meg a 3D modell.

A modell az egyetem tervrajzai alapján készült el. A feldolgozáshoz használt segédeszköz a Blender, ingyenes modellező software volt. A 3D-s modell ezután exportálásra kerül egy glTF (Grafikus nyelvátviteli formátum) kiterjesztésű fájlba, majd a modell komponenes ezt a filet felhasználva jeleníti meg a böngészőben az épület 3D-s reprezentációját.

6.2. Webalkalamzás architektúra

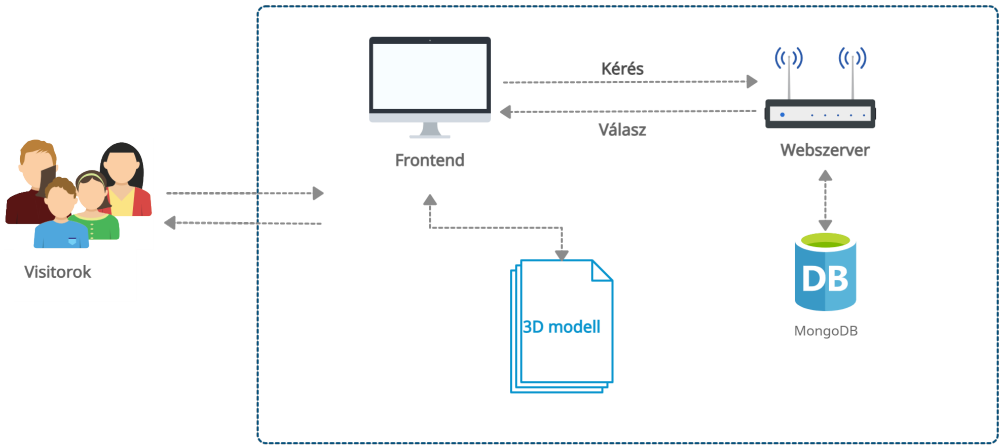
A tervezés során kideült, hogy négy nagy komponensre lesz szükség: Frontend, Webszerver, Adatbázis és a 3D modell.

Amint a 6.2 ábrán is látszik a Visitorok a Frontenden keresztül érik el az alkalmazást. A Frontend fogja bíztosítani a felhasználói felületet, vagyis itt fognak megjelenni az adatok az adatbázisból és a 3D modell is ide fog betöltődni.

A Webszerver segítségével lesznek elérhetőek a Frontend számára az adatbázisban tárolt adatok. Tulajdonképpen a Webszerver biztosít egy kommunikációs csatornát a Frontend és az adatbázis között. A Frontend kéréseket (GET, PUT, POST) küld a webszervernek és a webszerver a kérésnek megfelelő eredményt szolgáltatja vissza.

Az adatbázis fogja tárolni a Frontendről érkező adatokat, valamint vissza is szolgáltatja azokat a megfelelő kérések esetén a Webszerveren keresztül.

A 6.1 fejezetben említett 3D modell egy statikus fáljban lesz elhelyezve, amelyet a Frontend ér el és onnan fogja betölteni a modellt egy Frontenden belüli komponensbe. A statikus fájl tartalmazza azokat az elemeket, amelyek megjelennek az alkalmazáson belül, viszont az adatbázisban nem lehetséges vagy nagyon nehéz eltárolni. Ilyen elem a 3D modell is mivel a mérete meghaladja a 30 MB. A Frontenden belüli komponensen keresztül fog összekapcsolódni a 6.1 fejezetben említett két alrészleg, maga a webapplikáció és a 3D modell.



6.2. ábra. *A webalkalmazás rendszer architektúrája*

6.3. A fejlesztéshez használt technológiák

A rendszer egy webalkalmazásként lett fejlesztve. Az alkalmazás fejlesztéséhez első sorban szükséges volt egy operációs rendszer amely támogatja a Vue.js-t, Spring Boot-t, MongoDB-t.

6.3.1. Frontend

A Frontend rész HTML/JavaScript-ben implementálódott, keretrendszernek pedig a Vue.js volt használva, mivel egy új, progresszív keretrendszer ezért, folyamatosan új dolgok használatát biztosítja a fejlesztők számára, így a keretrendszerek használatának felmérési listáján is elől szerepel. A Vue.js használatához a következő telepítéseket, utasításokat, parancsokat kellett végrehajtani ebben a sorrendben:

- nodejs és npm letöltése és installálása : segítségével könnyen tudjuk telepepíteni a Vue-t.
- npm install vue: a Vue telepítésének parancsa.
- vue init webpack sapi3dtour: a projekt inicializása.
- npm install vue-router: segített a routolás megoldásában.

- vue add vuetify - protopype: változatosabb html tageket add hozzá a projekthez.
- npm install @fortawesome/fontawesome-free: a *Font Awesome*[1] ikonok használatát tette lehetővé.
- npm install –save axios: segítségével tudtam kérni és küldeni adatokat a Backendre.
- npm install vue-3d-model –save: a 3D modell betöltéséért felelős csomag.

A Frontend indításához először bele kell lépni a sapi3dtour könyvtárba, ott nyitni egy console ablakot és lefutattni a következő parancssort: *npm run dev*.

6.3.2. Backend

A Backend rész Java-ba lett implementálva, keretrendszernek a Spring Bootot használtam, mivel a segítségével nem kell külön foglalkozni a Maven vagy Gradle fájlok, Tomcat szinkronizálásával. Ez annak köszönhető, hogy a Spring Boot fejlesztői kifejlesztettek egy weboldalt, ahol Spring Boot típusú projekteket lehet létrehozni. Ez az oldal nem csak a projekt strukturájának létrehozásában segít, hanem a különböző dependeciákat is betudjuk állítani egyszerűen. A Spring Boot projekt létrehozásához a következő lépéseket végeztem el:

- A legújabb Java (15.0.1) instalálása
- Spring Boot keretrendszer telepítése
- *Spring Initializr*[2] segítségével létre hozni a projektet különböző dependenciák megadásával.
- A következő dependenciákat adtam hozzá:
  - MongoDB: az adatbázis elérésében segít.
  - Spring Web: lehetővé teszi a Cors filterek beállítását.
  - Spring Security: a beépített autetikációt teszi lehetővé.

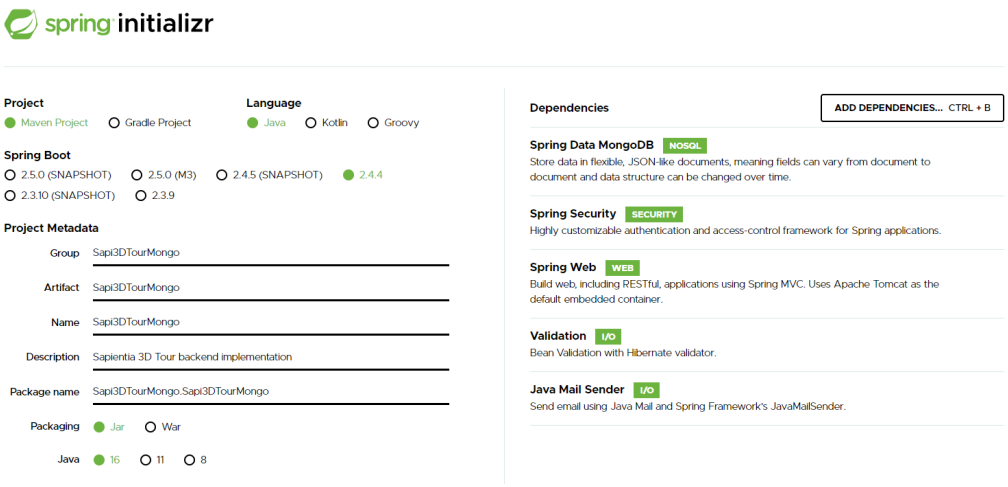
[1] https://fontawesome.com/icons?d=gallery&p=2&m=free

[2] Spring Initializr (a név helyesen van leírva): https://start.spring.io/

– Validation: a beérkező adatok validálásával foglalkozik.

– Java Mail Sender: elősegíti az e-mail küldést.

Az 6.3 ábrán látható egy példa a Spring Initializr használatára. A példa pontosan a fent leírtakat mutatja be.



6.3. ábra. *Példa a Spring Initializr használatára*

6.3.3. Adatbázis

Az adatbázis MongoDB NoSql lett, amelyet az operációs rendszernek megfelelő Workbench-ben kezeltem. Mivel az adatokat nem lehet egy séma alapján felállítani ezért van szükség a NoSQL adatbázisra.

Az adatbázis létrehozásához a következő lépéseket végeztem el:

- Telepítettem a MongoDB adatbázis kezelő rendszert.
- A MongoDB-hez tartozó Workbench telepítése
- Létrehoztam az adatbázist a Workbench segítségével
- Kollekciókat nem itt hoztam létre. Az a Spring Boot feladata.

6.4. Modulok leírása

6.5. Adatbázis

6.6. Verziókövetés

A rendszer fejlesztéséhez szükséges volt egy verziókövető rendszert választani. Erre azért volt szükség, mivel ketten készítettük a projektet, a verziókövető rendszer segítséget nyújtott, a sikeres közös fejlesztéshez. A verziókövető rendszer mellett egy projekt menedzsment eszközt is választottunk, ahol a feladatainkat (task) tudtuk nyilvántartani.

6.6.1. Bitbucket

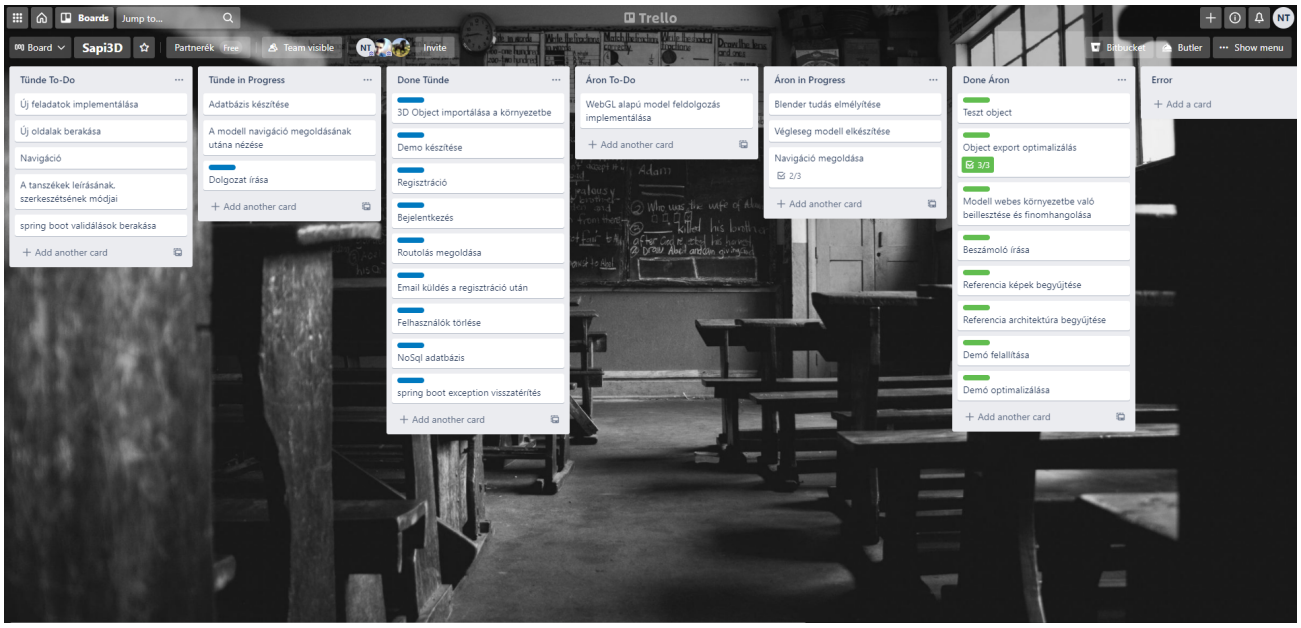
Kódverzió követő rendszer [21], amelyet a fejlesztők számára hoztak létre. Lehetőséget biztosít a git adattárak kezeléséhez és a fejlesztési folyamat végigvezetéséhez.

6.6.2. Trello

Egy projekt menedzsment eszköz [22], amely a projekteket különböző táblákba rendezi. A 6.4 ábrán látható egy példa az eszköz használatára. Megfigyelhetőek a különböző táblák, amelyekben láthatunk kissebb feladatokat.

A táblák tulajdonképpen a feladat folyamatát jelzi, mint például a To-Do tábla jelenti azokat a feladatokat amelyeket még el sem kezdtek a fejlesztők. A Progress tábla jelzi, hogy mely feladatokon dolgoznak a fejlesztők. A Done tábla jelzi, hogy mely feladatok lettek elvégezve. A fejlesztők eltudják látni a feladatokat egy-egy színnel is, amely jelzi a többi fejlesztőnek, hogy azt a feladatot ki végzi. A táblák között a feladatokat lehet hordozni.

Egy jó tulajdonsága a Trellonak, hogy össze lehet kötni a Bitbucket-tel így nem kell külön megnyítani a két felületet, hanem egy felület alatt meglehet nézni mindkettőt.



6.4. ábra. *Példa a Trello projekt menedzsment eszközre*

7. fejezet

Megvalósított rendszer

7.1. Megvalósítások

8. fejezet

Összefoglalás

8.1. Továbbfejlesztési lehetőségek

Irodalomjegyzék

[1] T. P. Kersten, F. Tschirschwitz, and S. Deggim, „Development of a virtual museum including a 4d presentation of building history in virtual reality," *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 42, p. 361, 2017.

[2] D. Dedov, M. Krasnyanskiy, A. Obukhov, and A. Arkhipov, „Design and development of adaptive simulators using 3d modeling," *International Journal of Applied Engineering Research*, vol. 12, no. 20, pp. 10415–10422, 2017.

[3] R. K. Moloo, S. Pudaruth, M. Ramodhin, and R. B. Rozbully, „A 3d virtual tour of the university of mauritius using webgl," in *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, pp. 2891–2894, IEEE, 2016.

[4] D. Perdana, A. I. Irawan, and R. Munadi, „Implementation of a web based campus virtual tour for introducing telkom university building," *International Journal if Simulation—Systems, Science & Technology*, vol. 20, no. 1, pp. 1–6, 2019.

[5] R. Napolitano, I. Douglas, M. Garlock, and B. Glisic, „Virtual tour environment of cuba's national school of art," *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci*, vol. 42, no. 2, p. W5, 2017.

[6] C. Maines and S. Tang, „An application of game technology to virtual university campus tour and interior navigation," in *2015 international conference on developments of E-systems engineering (DeSE)*, pp. 341–346, IEEE, 2015.

[7] „Database Management System Tutorial." https://www.tutorialspoint.com/dbms/index.htm.

[8] S. Venkatraman, K. Fahd, S. Kaspi, and R. Venkatraman, „Sql versus nosql movement with big data analytics," *Int. J. Inform. Technol. Comput. Sci*, vol. 8, pp. 59–66, 2016.

[9] A. Gupta, S. Tyagi, N. Panwar, S. Sachdeva, and U. Saxena, „Nosql databases: Critical analysis and comparison," in *2017 International Conference on Computing and Communication Technologies for Smart Nation (IC3TSN)*, pp. 293–299, IEEE, 2017.

[10] P. Martins, M. Abbasi, and F. Sá, „A study over nosql performance," in *World Conference on Information Systems and Technologies*, pp. 603–611, Springer, 2019.

[11] „Frameworks." https://hackr.io/blog/what-is-frameworks.

[12] „Top 10 emerging backend frameworks in 2021." https://www.decipherzone.com/blog-detail/top-10-backend-development-frameworks.

[13] „The Most Popular Backend Frameworks for Web Development in 2020." https://www.intagleo.com/blog/most-popular-backend-frameworks-for-web-development-in-2019/.

[14] E. Wohlgethan, *SupportingWeb Development Decisions by Comparing Three Major JavaScript Frameworks: Angular, React and Vue. js*. PhD thesis, Hochschule für Angewandte Wissenschaften Hamburg, 2018.

[15] „Vue and Angular comparison." https://www.educative.io/blog/react-angular-vue-comparison.

[16] N. Js and N. JS, „Node. js," *Tradução de: SILVA, AG Disponível em*, 2016.

[17] „NodeJs vs Spring Boot." https://www.chapter247.com/blog/node-js-vs-springboot-java-which-one-to-choose-and-when/.

[18] „Spring Framework." https://spring.io/projects/spring-framework.

[19] Ž. Jovanović, D. Jagodić, and Vujičić, „Java spring boot rest web service integration with java artificial intelligence weka framework," in *International Scientific Conference "UNITECH 2017*, pp. 323–327, 2017.

[20] „NodeJS vs Spring Boot : Picking up the right Technology.” https://www.inexture.com/nodejs-vs-spring-boot-choosing-the-best-technology/.

[21] „Bitbucket: What is Bitbucket?.” https://confluence.atlassian.com/confeval/development-tools-evaluator-resources/bitbucket/bitbucket-what-is-bitbucket.

[22] „What is Trello?.” https://help.trello.com/article/708-what-is-trello.