2. Követelmény, projekt, funkcionalitás

2.1 Bevezetés

2.1.1 Cél

A dokumentum célja, hogy ismertesse a projekt követelményeit és funkcionalitását. Ezek segítségével egy kész szoftver előállítása a projekt végső célja.

2.1.2 Szakterület

A szoftver egy szórakoztatás céljából készült videójáték. Vasúthálózatok kiépítésére és fenntartására alkalmas szimulátor, mely a felhasználó stratégikus képességeit méretteti meg.

2.1.3 Definíciók, rövidítések

***MVC**: Model-View-Control*

***Szkeleton**: váz, megíratlan metódusokkal*

***Proto**: kész metódusokkal rendelkezik, de nincs grafikus felület még*

***HSZK**: Hallgatói Számítógép Központ*

***UML**: Unified Modeling Language*

***JRE**: Java Runtime Environment - futtatókörnyezet*

***Eclipse**: fejlesztő környezet*

***GIT**: verziókezelő*

***GitLab**: verziókezelő rendszerre épülő internetes szolgáltatás*

***GitHub**: verziókezelő rendszerre épülő internetes szolgáltatás és kliens*

***IIT**: Irányítástechnika és Informatika Tanszék*

***Skype**: Egy alkalmazás, ami lehetővé teszi az internetes hangalapú kommunikációt*

***Verziókezelő**: egy olyan szolgáltatás, aminek segítségével el lehet tárolni a régebbi verzióit a fájloknak, így gond esetén bármikor vissza lehet térni egy-egy régebbi verzióhoz*

2.1.4 Hivatkozások

***Java** - https://www.java.com*
***IIT** - https://www.iit.bme.hu*
***Lego Loco** - https://en.wikipedia.org/wiki/Lego_Loco*
***Traintown Deluxe** - https://en.wikipedia.org/wiki/3D_Ultra_Lionel_Traintown*

2.1.5 Összefoglalás

A dokumentum további részei ismertetni fogják a projekttel kapcsolatos további terveket. Megtalálhatók a különféle funkciók és szoftverkövetelmények, illetve a projektnapló.

2.2 Áttekintés

2.2.1 Általános áttekintés

A kialakítandó szoftver legmagasabb szintű architekturális képe a Model-View-Control (MVC). E három alrendszerből fog felépülni a szoftver. A Model alkotja a vázat, melyben a legfontosabb eljárások találhatók. A View alrendszer segítségével a program grafikusan fogja megjeleníteni az adatokat, a Control pedig a felhasználói interakciókat kezeli. Ezek képezik a felhasználói kapcsolatok alapját. Hálózati hozzáférést a szoftver nem igényel. Adattárolási szempontból a szoftver képes lesz a játékmenet adott pillanatát fájlba elmenteni, és ezt egy későbbi futás során akár visszatölteni.

2.2.2 Funkciók

*A **program** egy valós idejű vonatos stratégiai játék. A játék lényege, hogy az előre elkészített játékpályákon, azaz úgynevezett **terepasztalokon** vonatok jelennek meg látszólag véletlen időközönként a **bejövő távolsági sínpárokon.** A vonatokat a játékosnak kell terelnie váltók segítségével a megfelelő sínekre, amikkel el tudja juttatni az állomásokra az utasokat. Ezen kívül használhat úgynevezett alagutakat is.*

*A **terepasztal** a játék alapja, ezen található minden sín, alagút, vonat, állomás, váltó.*

*Az **alagutak** arra használhatók, hogy megépítésük után a vonat bármelyik oldalról belemehet, és a megépített alagút túloldalán jön ki belőle, ezzel létrehozva egy névleges új sínt. Az alagutat csak bizonyos helyekre lehet építeni a pályán, azaz csak olyan sínszakaszokra, amik meg vannak jelölve, hogy alagútépítésre alkalmasak. Az alagút építésének folyamata úgy működik, hogy először a játékos elhelyezi az egyik végét, ekkor még csak tervben van az építkezés, azaz az alagút még nem funkcionál, a vonatok elmennek mellette, és nem mennek bele. Amint lerakjuk a másik végét az alagútnak, megépül, és használhatóvá válik a vonatok számára. Az alagút lebontása hasonló elven működik, bármelyik végét lebonthatjuk, ezáltal megszűnik a funkcionalitása, azaz mivel csak egyik alagút bejárat van megépítve, így a vonatok nem mennek az alagútba. A Sugár Állami Vasutak (SÁV) balesetmentes közlekedési tervezetének köszönhetően a balesetek elkerülésének érdekében az alagút nem bontható, amíg vonat tartózkodik benne. Az alagút hossza a bejáratok távolságával egyenesen arányos.*

*A **váltók** ennél egyszerűbben működnek, itt a játékos egyszerűen átkapcsolhatja a váltót, ezáltal módosítva a lehetséges pályáját a vonatoknak. A váltó nem kapcsolható, ha egy vonat éppen halad át rajta, és a SÁV nem is javasolja, hogy megpróbáljuk. Amennyiben egy vonat a váltó inaktív ágáról érkezik, a váltó automatikusan át fog állni. A **sínek** működésük szempontjából elég egyszerűek, mindkét végükkel kapcsolódnak vagy egy másik sínhez, vagy a bejövő távolsági sínpárokhoz, illetve kapcsolódhatnak még váltókhoz és alagútbejáratokhoz is. Ha ezek bármelyikén egynél több vonat tartózkodik, akkor azok felrobbannak, és elvesztjük a játékot.*

*A **bejövő távolsági sínpárok** olyan különleges sínpárok, ahonnan érkezhetnek vonatok. Több is lehet belőlük a terepasztalon, így nehezítve a játékos dolgát, megosztva a figyelmét.*

*A **vonatok** egy mozdonyból és egy vagy több vasúti kocsiból állnak. Számuk a pálya nehézségétől függ, és látszólag véletlen időközönként érkeznek a bejövő távolsági sínpárokon.*

*A **mozdony** minden vonat elemi része, a vasúti kocsik elején található minden esetben, azaz nem foglalkozunk tolatómozdonyos vonatokkal. Nem utaznak utasok a mozdonyban, a mozdonyvezető csak az után hagyja el a mozdonyt, hogy megnyertük a játékot és már nem látjuk.*

*Az **állomások** a sínek mellett találhatóak közvetlenül, és színük is van. Amennyiben egy*

ugyanilyen színű vasúti kocsi érkezik az állomáshoz, és a vasúti kocsi előtt található kocsik már kiürültek, akkor ez a kocsi is kiürül, mégpedig a SÁV balesetvédelmi előírásainak megfelelően leugranak az utasok a vonatról.

*A **játék célja** az, hogy a játékos az összes pályán lévő vonatnak a vasúti kocsijait kiürítse. Ezt úgy tudja elérni, hogy a váltókat kapcsolgatja és alagutakat épít, ezáltal a megfelelő állomásokhoz juttatva a vonatokat, miközben arra is figyel, hogy ne ütközzenek. Amint az összes vonat összes kocsija üres, a játékos nyert. Amennyiben bármelyik vonat ütközik egy másikkal, a játékos vesztett.*

*Amennyiben a játékos **megnyerte** az adott pályát, a következő pályára lép és győzelmének ideje rögzítésre kerül az **eredményjelzőn,** ahol minél gyorsabban sikerült megoldania a pályát, annál jobbnak számít az ideje.*

2.2.3 Felhasználók

A felhasználó nagyon széles körű spektrumból jöhet, hiszen a videójátékok témaköre sok embert érint. Előismeretek nélkül is könnyen megtanulható a szoftver. Feltételezhető, hogy alapszinten ért a számítógép kezeléséhez, mint például egérmozgatás és kattintás.

2.2.4 Korlátozások

Előírás, hogy a szoftvert java nyelven kell írni, és az alapvető könyvtárakon kívül semmilyen más külső könyvtár nem használható.

2.2.5 Feltételezések, kapcsolatok

***Java** - https://www.java.com*
A fejlesztéshez használt programnyelv weboldala.
***IIT** - https://www.iit.bme.hu*
A megrendelő weboldala, a feladatkiírás itt olvasható.
***Lego Loco** - https://en.wikipedia.org/wiki/Lego_Loco*
98-as videójáték, mely inspirációként szolgál és segít a tervezésben.
***Traintown Deluxe** - https://en.wikipedia.org/wiki/3D_Ultra_Lionel_Traintown*
99-es videójáték, szintén ötletmerítésre

2.3 Követelmények

2.3.1 Funkcionális követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Use-case	Komment
1.01	*a szélekről vonatok indulnak*	*bemutatás*	*alapvető*	*megrendelő*		
1.02	*a vonatban egy mozdony és kocsik*	*bemutatás*	*alapvető*	*megrendelő*		*Egy darab mozdony és mögötte a kocsik*
1.03	*a vonatok sínen közlekednek*	*bemutatás*	*alapvető*	*megrendelő*		
1.04	*elágazásoknál a váltók állíthatók*	*bemutatás*	*alapvető*	*megrendelő*	*Váltó állít*	*rá kell kattintani*
1.05	*két alagútbejárat építhető és rombolható*	*bemutatás*	*alapvető*	*megrendelő*	*Alagút épít*	*vagy mindkettő aktív, vagy egyik sem*
1.06	*a kocsik sorban ürülnek ki, színek szerint a megállóknál*	*bemutatás*	*alapvető*	*megrendelő*	*Állomáshoz ér*	*győzelem*
1.07	*két vonat összeütközhet*	*bemutatás*	*alapvető*	*megrendelő*		*veszítés*
1.08	*pálya teljesítéskor az állás elmentődik*	*bemutatás*	*opcionális*	*csapat*		
1.09	*váltó átváltódik ha nem abba az irányba áll amerről a vonat jön*	*bemutatás*	*alapvető*	*csapat*	*Váltó állít*	
1.10	*jelzés, ha vonat érkezik a pályára*	*bemutatás*	*fontos*	*csapat*		*ütközések elkerülésére*
1.11	*vonat visszajön, ha kimegy a pályáról*	*bemutatás*	*fontos*	*csapat*		

2.3.2 Erőforrásokkal kapcsolatos követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Komment
2.01	*Git*	*nincs*	*alapvető*	*csapat*	*verziókezelő*
2.02	*Gitlab repo*	*nincs*	*alapvető*	*csapat*	*tárhely*
2.03	*Eclipse*	*nincs*	*opcionális*	*csapat*	*fejlesztő környezet*
2.04	*WhiteStarUML*	*nincs*	*opcionális*	*csapat*	*UML diagramszerkesztő*
2.05	*JRE*	*bemutatás*	*alapvető*	*megrendelő*	*futtató környezet*
2.06	*elég teljesítményű hardver*	*bemutatás*	*alapvető*	*megrendelő*	*elég: HSZK gépei (monitor, egér is)*

2.3.3 Átadással kapcsolatos követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Komment
3.01	*Szkeleton bemutatása március 22-én*	*bemutatás*	*alapvető*	*megrendelő*	
3.02	*Proto bemutatása április 19-én*	*bemutatás*	*alapvető*	*megrendelő*	
3.03	*Végleges bemutatás május 10-én*	*bemutatás*	*alapvető*	*megrendelő*	*grafikus felülettel*

2.3.4 Egyéb nem funkcionális követelmények

Nincsenek egyéb nem funkcionális követelmények.

2.4 Lényeges use-case-ek

2.4.1 Use-case leírások

Use-case neve	Váltó állít
Rövid leírás	A felhasználó állítani tudja a váltókat.
Aktorok	User
Forgatókönyv	A felhasználó rákattint a váltóra, így a sín átáll egy másik irányba, ezzel módosítja a vonat útját.

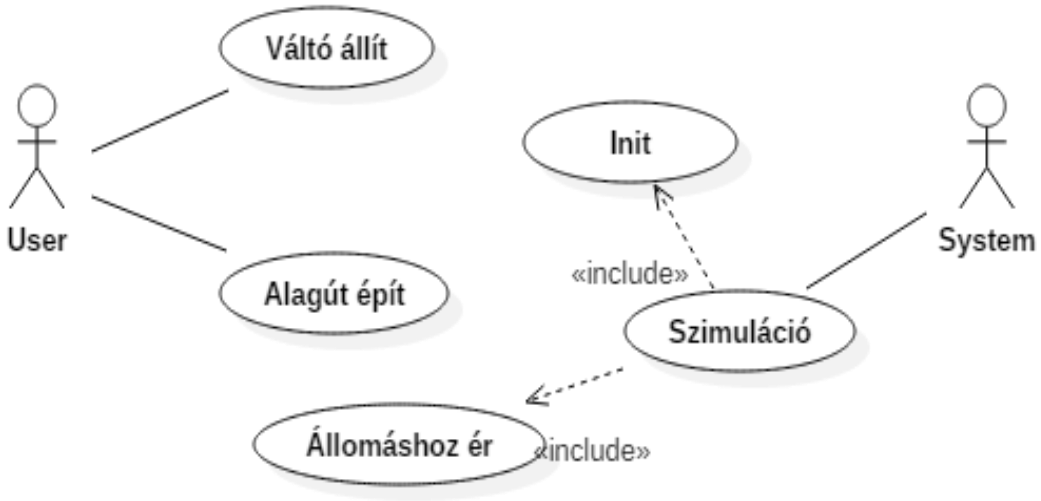
Use-case neve	Alagút épít
Rövid leírás	A felhasználó új alagutakat tud hozzáadni.
Aktorok	User
Forgatókönyv	A felhasználó először az alagút egyik végét, majd a másikat kell, hogy megépítse. Ezután a vonatok képesek az alagút két vége között közlekedni.

Use-case neve	Szimuláció
Rövid leírás	A vasúthálózaton a váltóknak és az alagutaknak megfelelően vonatok közlekednek.
Aktorok	System
Forgatókönyv	A vonatok a síneken és az alagutakban közlekednek. Továbbá az init és az Állomáshoz ér include-okon keresztül valósul meg.

Use-case neve	init
Rövid leírás	A vasúthálózat, váltók, sínek, vonatok létrehozása, elindítása.
Aktorok	System
Forgatókönyv	A vasúthálózatok egy előre megírt fájlból lesznek betölthetőek.

Use-case neve	Állomáshoz ér
Rövid leírás	A vonat elhalad az állomás mellett.
Aktorok	System
Forgatókönyv	A vonatról leugranak az utasok az adott kocsi színének megfelelően.

2.4.2 Use-case diagram



2.5 Szótár

Alagútépítés: *Kattintással építhető a bejárat. Akkor épül meg az alagút, ha már mindkét bejárata megépült.*

Alagút: *Sínek összekötésére és ütközések elkerülésére szolgáló építmény, egy darab található belőle a terepasztalon.*

Alagútbejárat: *Az alagúthoz két bejárat tartozik, ezeket a játékos építheti meg.*

Állomás: *A terepasztalon a sínek mellett található. Színe alapján szállnak le az utasok.*

Bejövő távolsági sínpár: *A vonat ezen érkezik a terepasztalra.*

Cél: *Az összes vonat kiürítése, az utasok megfelelő állomásra eljuttatása ütközés nélkül.*

Eredményjelző: *Minden pályához tartozik egy, ezen sorrendben szerepelnek az adott pályát legeredményesebben teljesítő játékosok.*

Győzelem: *A cél elérése vesztés nélkül.*

Játékos: *Az ember, aki játszani akar a játékkal.*

Kocsi: *Utasokat szállít szín szerint. Kiürül, ha azonos színű állomáson halad át és nincs előtte teli kocsi.*

Mozdony: *A vonat elején helyezkedik el, nincs színe*

Összeütközés: *Két vonat egymással szembe kerül és egymásba rohannak. Ekkor a játékos elveszti a játékot.*

Sín: *Ezeken közlekednek a vonatok.*

Terepasztal: *ezen helyezkednek el az állomások, sínek és vonatok.*

Utas: *Abban a színű kocsiban ül, amelyik színű állomáson le szeretne szállni. Cél, hogy elszállítsunk minden utast.*

Váltó állítás: *Kattintással állítható át a másik sínpárra.*

Váltó: *Vonatok átirányítására szolgál.*

Vesztés: *Két vonat összeütközése esetén.*

Vonat: *egy darab mozdonyból és legalább egy darab kocsiból áll.*

2.6 Projekt terv

2.6.1 Ütemterv

Dátum	Feladat
február 20.	Követelmény, projekt, funkcionalitás - beadás
február 27.	Analízis modell kidolgozása 1. - beadás
március 6.	Analízis modell kidolgozása 2. - beadás
március 13.	Szkeleton tervezése - beadás és a dokumentum herculesre való feltöltése.
március 20.	Szkeleton - beadás és a forráskód herculesre való feltöltése
március 27.	Prototípus koncepciója - beadás
április 3.	Részletes tervek - beadás
április 10.	Prototípus készítése, tesztelése
április 17.	Prototípus - beadás és a forráskód, a tesztbemenetek és az elvárt kimenetek herculesre való feltöltése
április 24.	Grafikus felület specifikációja - beadás
május 1.	Grafikus változat készítése
május 8.	Grafikus változat - beadás és a forráskód herculesre való feltöltése
május 10.	Összefoglalás - beadás és feltöltés

2.6.2 Csapat

Név	Feladat
Dócs Zoltán	Kód, dokumentáció, naplózás
Krátky Gergő Ádám	Kód, dokumentáció, dokumentáció organizálása
Sillye Márk	Kód, dokumentáció, visual designer
Szili Péter	Kód, dokumentáció, nyomtatás, GitLab organizálása
Varga János	Kód, dokumentáció, UML

2.6.3 Kommunikáció

Alapvetően kommunikációt élőben vagy Facebook chat-en folytatjuk, emellett hetente minimum egyszer csapat megbeszélést tartunk.

Csapat megbeszélésekhez Skype konferenciahívást használunk.

Dokumentáció megosztása, írása a Google Docs-ban és az Online Office-ban történik. A dokumentumokhoz mindenkinek teljes hozzáférése van.

Verziókezeléshez GitLab-ot használunk, GitHub klienssel.

2.7 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2017.02.12. 15:14	1 óra	Dócs Krátky Sillye Szili Varga	Projekt előkészítése, ötletelés, feladat kiírás átnézése
2017. 02. 18. 11:00	2 óra	Dócs Krátky Sillye Szili Varga	Csapat megbeszélés, dokumentáció készítése
2017. 02. 18. 13:30	3,5 óra	Dócs Krátky Sillye Szili Varga	Dokumentáció szerkesztése
2017. 02. 19. 19:45	0,5 óra	Krátky	Dokumentáció átolvasása
2017. 02. 19. 20:50	0,5 óra	Szili	Dokumentáció átolvasása
2017. 02. 19. 20:15	0,5 óra	Varga	Dokumentáció átolvasása
2017. 02. 19. 21:00	0,5 óra	Dócs	Dokumentáció átolvasása
2017. 02. 19. 22:00	0,5 óra	Sillye	Dokumentáció átolvasása