



Karelia-ammattikorkeakoulu
Tradenomi, Tietojenkäsittely (AMK)

Argumentointiin perustuva pelimekaniikka

Niko Kekkonen

Opinnäytetyö, joulukuu 2023

www.karelia.fi



OPINNÄYTETYÖ
Joulukuu 2023
Tietojenkäsittelyn koulutus

Tikkarinne 9
80200 JOENSUU
+358 13 260 600

Tekijä
Niko Kekkonen

Nimeke
Argumentointiin perustuva pelimekaniikka

Tiivistelmä

Opinnäytetyön tavoitteena oli luoda peli, jonka pelimekaniikkana on argumentointi. Pelin lähtötilanne on, että pelaaja haluaa tehdä elämässään jonkun muutoksen. Pelihahmot vastustelevat pelaajaa tekemästä muutosta ja pelaaja yrittää saada pelihahmot hyväksymään muutoksen. Pelihahmojen vastaukset generoidaan ChatGPT-keskustelubotin avulla. Teoriaosuudessa käydään läpi pelimekaniikan käsitteitä, kuinka luoda hyvä pelimekaniikka ja pelimekaniikkoja videopeleissä.

Peli toteutettiin Unreal Engine 5 -pelimoottorilla ja ChatGPT-integraatio OpenAI API:n avulla C++-ohjelmointikieltä käyttäen. Muut toiminnallisuudet, kuten liikkuminen, kentät ja viesti-ikkuna toteutettiin Unreal Enginen blueprinttien avulla.

Argumentointiin perustuvan pelimekaniikan luominen onnistui hyvin ChatGPT-integraation avulla. Pelihahmot vastustelevat pelaajan muutosta ja pelaajan täytyy saada pelihahmot hyväksymään muutos. Pelimekaniikkaa voi hyödyntää esimerkiksi apuna päätöksenteossa tai yleisesti hyödyntää ChatGPT:tä pelihahmojen vastauksien generoimisessa.

Kieli
suomi

Sivuja 40
Liitteet 0
Liitesivumäärä 0

Asiasanat
Unreal Engine 5, pelimekaniikka, ChatGPT



THESIS
December 2023
Degree Programme in Business Information Technology

Tikkarinne 9
80200 JOENSUU
FINLAND
+ 358 13 260 600

Author
Niko Kekkonen

Title
Game Mechanics Based on Argumentation

Abstract

The purpose of this thesis was to create a game with a game mechanic based on argumentation. The starting point of the game is that the player wants to make a change in their life. The game characters oppose the player to make the change and the player tries to get the game characters to accept the change. The responses of the game characters are generated using ChatGPT chatbot. In the theory section, the concept of game mechanics are introduced and how good game mechanics and game mechanics in video games can be implemented.

The game was made with Unreal Engine 5 game engine and the ChatGPT integration with the OpenAI API using C++ programming language. The functionalities, such as movement, levels and message windows were implemented with Unreal Engine blueprints.

Creating a game mechanic based on argumentation was successful with the ChatGPT integration. The game characters oppose the player's change, and the player must make the game characters accept the change. This game mechanic can be used, for example, as a help in decision-making, or in general you can use ChatGPT to generate responses of game characters.

Language
Finnish

Pages 40
Appendices 0
Pages of Appendices 0

Keywords
Unreal Engine 5, game mechanics, ChatGPT

Sisältö

1	Johdanto	5
2	Pelimekaniikat videopeleissä	6
2.1	Pelimekaniikka yleisesti	6
2.2	Pelitila	7
2.3	Pelimekaniikan jaottelu	7
2.4	Pelimekaniikan päämäärä	8
2.5	Yleisiä mekaniikkoja peleissä	10
2.5.1	Elämäpisteet	10
2.5.2	Liikkuminen	11
2.5.3	Pelin sisäinen talous	12
2.5.4	Keskustelu	13
2.5.5	Listaus erilaisista pelimekaniikoista	13
3	Työkalut	15
3.1	Unreal Engine 5	15
3.2	OpenAI API	15
4	Pelin toteutus ja sen vaiheet	16
4.1	Suunnitelma	16
4.2	Tekninen toteutus	17
4.3	Lopputuotos	27
4.4	Keskustelu pelihahmon kanssa	30
5	Pohdinta	34
5.1	Argumentointiin perustuvan pelimekaniikan arviointi	34
5.2	ChatGPT:n käyttö pelihahmojen vastausten generoimisessa	36
5.3	Argumentointiin perustuvan pelimekaniikan hyödyntäminen	36
5.4	Tekninen toteutus	37
5.5	Vertailu AIDungeoniin	37
5.6	Jatkokehitys	38
	Lähteet	39

1 Johdanto

Opinnäytetyön aiheena on argumentointiin perustuva pelimekaniikka ja tavoitteena on luoda peli, jonka keskeinen pelimekaniikka on argumentoida pelihahmojen kanssa. Pelihahmojen vastaukset generoidaan ChatGPT-keskustelubotin avulla pelaajan vastausten perusteella.

Pelin lähtötilanne on, että pelaaja haluaa tehdä elämässään jonkun muutoksen, jonka pelaaja määrittelee pelin alussa. Pelaajan määrittelemä asia, jota hän haluaa muuttaa elämässään, toimii keskustelun aiheena ChatGPT-keskustelubotille. Peli itsessään on kävelysimulaattori, jossa pelaaja kohtaa pelihahmoja. Jokaiseen pelihahmoon on integroitu ChatGPT-sessio, joka tuottaa vastaukset pelaajalle. Pelihahmot vastustelevat pelaajaa tekemästä muutosta elämässään ja pelaaja yrittää saada pelihahmot hyväksymään pelaajan ennalta määrätyn muutoksen pelaajan elämässä. Kun kaikki pelihahmot on käyty läpi ja ne ovat hyväksyneet muutoksen, on pelaaja voittanut pelin.

Muita ChatGPT:n tekstigenerointia hyödyntäviä pelejä on jo olemassa, kuten AI Dungeon (2019). AI Dungeon on tekstipohjainen fantasisimulaatio, jossa pelin tarina ja hahmot luodaan täysin ChatGPT:n avulla pelaajan syötteiden perusteella. Pelaaja voi valita teeman tarinalle, kuten fantasia, mysteeri tai tehdä oman teeman tarinalle. (Latitude 2023.)

Pelimekaniikat videopeleissä -luvussa käyn läpi määritelmiä pelimekaniikasta. Seuraavissa luvuissa käyn läpi käytettyjä ohjelmistoja pelin luomiseen sekä vaiheita itse pelin kehityksestä.

2 Pelimekaniikat videopeleissä

2.1 Pelimekaniikka yleisesti

Pelimekaniikat ovat pelin tekijän ennalta määäämiä sääntöjä, jotka määäävät, kuinka pelaaja ja peli ovat vuorovaikutuksessa toistensa kanssa. Pelimekaniikat toimivat sääntöinä pelaajalle. Nämä säännöt kertovat, kuinka pelaaja pelaa peliä. Pelimekaniikat myös määäävät, kuinka peli itsessään vastaa pelaajan toimiin. Pelimekaniikkoja voi olla useita erilaisia ja pelin tekijä itse voi päättää, minkälaisista pelimekaniikoista peli koostuu. (Boller 2013.)

Pelimekaniikat ovat siis sääntöjä, joita pelaaja ja peli noudattavat. Sääntöjen tarkoituksesta ja käytöstä on kuitenkin erilaisia näkemyksiä. Daniel Cookin (2006) mukaan pelimekaniikat ovat sääntöpohjaisia järjestelmiä tai simulaatioita, jotka johdattelevat tai kannustavat pelaajaa tutkimaan tai oppimaan pelaajan mahdollisuuksia vastemekanismien avulla. James Griesemer (2011) näkee pelimekaniikat enemmän rajoittavana tekijänä. Hänen mukaansa pelimekaniikat rajoittavat vuorovaikutteisuutta siten, että se johdattaa pelaajaan hauskempaan kokemukseen. (Griesemer 2011.) Alexander Brazien (2022) mukaan pelimekaniikat ovat asioita, joilla on seurauksia pelissä. Hänen mukaansa jokin asia yhdessä pelissä voi olla pelimekaniikka ja toisessa ei. Hänen yhdessä esimerkissäänsä liekehtivän nuolen ampuminen köynnöksiin paljastaa polun pelaajalle, mutta toisessa pelissä liekehtivän nuolen ampuminen aiheuttaa vain efektin ilman mitään muita seurauksia. Tärkein ero näillä on, että toisessa tapauksessa pelaajaan teoilla on pelin kannalta merkittäviä seurauksia ja toisessa ei. (Brazie 2022.)

Pelimekaniikkaa voidaan siis määritellä useilla eri tavoilla. Oli se sitten rajoittava, kannustava tai seuraamuksellinen tekijä pelissä, voidaan pelimekaniikkaa pitää sääntöinä, joita peli ja pelaaja noudattavat. Nämä säännöt määrittävät, mitä pelaaja voi tehdä ja miten peli vastaa pelaajan toimiin pelissä.

2.2 Pelitila

Aki Järvisen (2008) mukaan pelit ovat järjestelmiä, jotka tuottavat erilaisia tiloja pelin edetessä. Esimerkkejä erilaisista tiloista käytännössä Järvisen mukaan on pisteitten muutos, haasteitten muutos tai pelaajan sijainti peliympäristössä. Pelaaja vastaa pelin tulosteeseen omalla syötteellään ja peli vastaa siihen tilalla missä peli on. Pelit tarvitsevat erilaisia tiloja, jotta sääntöjä voidaan hyödyntää tiettyjen tilanteitten ratkaisemiseksi. Nämä ovat tarpeellisia pelin kannalta, että peli voi jatkua. (Järvinen 2008, 50.) Pelaaja pyrkii vaikuttamaan pelin tilaan pelimekaniikkojen avulla saavuttaakseen päämääränsä (Järvinen 2008, 255).

Järvinen (2008) jaottelee pelin tilat vähäisiin ja tärkeisiin pelitiloihin. Tärkeissä pelitiloissa saavutetaan korkeamman tason päämäärä. Vähäisissä pelitiloissa taas saavutetaan alemman tason päämäärä, joka on kuitenkin tärkeää korkeamman tason päämäärän saavuttamisen kannalta. (Järvinen 2008, 53.)

2.3 Pelimekaniikan jaottelu

Kaikkia pelimekaniikkoja ei ole välttämättä ole luotu yhtä arvokkaiksi tai hyödyllisiksi keskenään. Pelimekaniikkoja voidaan jakaa eri luokkiin. Carlo Fabricatoren (2007) mukaan pelimekaniikat jaetaan hierarkkisiin luokkiin niiden tärkeyden mukaan pelaajalle. Pelimekaniikat jaetaan ydin-, ensisijaisiin ja toissijaisiin pelimekaniikkoihin (Sicart 2008).

Ydinpelimekaniikat ovat mekaniikkoja, joita pelaaja tarvitsee jatkuvasti läpäistäkseen pelin, päästääkseen pelin voittotilaan. Sicart (2008) kertoo esimerkissään, että hyppiminen ensimmäisen persoonan ammuntopelin monipelissä ei ole ydinpelimekaniikka. Vaikka pelaaja hyppii jatkuvasti pelissä, se ei välttämättä auta pelaajaa saavuttamaan pelin voittotilaa. Kaikissa peleissä ei kuitenkaan ole haluttua voittotilaa. Tällaisissa peleissä on kuitenkin ydinpelimekaniikkoja, vaikka voittotilaa ei olisikaan. Voittotilan sijasta on jokin tila, johon pelaaja pyrkii pääsemään. Tähän tilaan pääsemiseen ydinpelimekaniikat auttavat. (Sicart 2008.)

Vaikka Sicart (2008) kertoo, että hyppiminen ensimmäisen persoonan ammuntopelissä ei ole ydinpelimekaniikka, voidaan hyppimistä pitää ydinpelimekaniikkana toisenlaisissa peleissä. Tasohyppelypelit etenkin videopelien lajityyppinä käyttävät hyppimistä ja liikkumista pelattavuuden luomiseksi (Klappenbach 2021).

Ensisijaiset pelimekaniikat ovat ydinpelimekaniikkoja, joita voidaan käyttää sellaisten haasteitten ratkaisemiseen, jotka johtavat haluttuun voittotilaan. Ensisijaiset pelimekaniikat ovat helposti saatavilla, ne on selitetty aikaisessa vaiheessa peliä ja niitä käytetään jatkuvasti pelin aikana. Nämä mekaniikat on asetettu pelaajalle ilmiselviin painalluksiin käytettävässä ohjaimessa, ne ovat aina saatavilla ja pysyvät muuttumattomana koko pelin ajan. Toissijaiset pelimekaniikat ovat ydinpelimekaniikkoja, jotka helpottavat saavuttamaan voittotilan. Toissijaiset mekaniikat eivät ole aina saatavilla tai vaativat ensisijaista mekaniikka toimiakseen. (Sicart 2008.)

2.4 Pelimekaniikan päämäärä

Pelimekaniikan tarkoitus on mahdollistaa vuorovaikutus pelin ja pelaajan välillä. Pelimekaniikkoja on useita, kuten on niihin lähestymistapojakin. Vaikka puhuttaisiin tietyistä pelimekaniikasta peleissä, eroavaisuuksia tietyssä pelimekaniikassa esiintyy pelistä toiseen. Tietyn pelimekaniikan toteuttamisen lähestymistapaan vaikuttaa se, mitä halutaan saavuttaa. On kuitenkin asioita, joita hyvän pelimekaniikan luomiseen kannattaa ottaa huomioon. Polydin Studio (2023) on koonnut listan seikoista, joista mukaansatempaava ja miellyttävä pelimekaniikka koostuu pelattavuuden kannalta.

- Selkeä tavoite

Selkeällä tavoitteella ohjataan pelaajan toimintaa ja päätöksiä pelissä. Pelaajalle selkeällä mekaniikalla saadaan pelaaja ymmärtämään mitä pelimaailmassa täytyy saavuttaa sekä päämäärän, eli halutun pelitilan.

- Haaste ja palkitsevuus

Haastava ja palkitseva mekaniikka vaatii pelaajalta taitoa ja strategiaa selviytyä haasteista pelissä. Palkitsemalla pelaaja hyvästä suorituksesta pelissä jollakin merkitsevällä asialla saa pelaajan jatkamaan peliä ja selviytymään uusista haasteista.

- Tasapaino

Tasapainoinen mekaniikka ei anna selkeää ylilyöntiasemaa millekään asialle pelissä tai pelaajalle varmistaen tasapainoisen pelikokemuksen. Etenkin moninpeleissä tasapaino on tärkeä, jotta yksittäinen pelaaja ei voi olla etulyöntiasemassa.

- Vuorovaikutus

Vuorovaikutus on keskeinen osa pelimekaniikkaa, pelaajan toimilla täytyy olla seurauksia. Seuraukset täytyy näkyä jotenkin pelimaailmassa ja vaikuttaa siihen.

- Valinta ja seuraus

Merkitykselliset valinnat ja seuraukset näille valinnoille antavat pelaajalle enemmän valtaa muodostaa haluamansa pelikokemus.

- Nousujohtoisuus

Nousujohteinen mekaniikka antaa pelaajan ymmärtää ja hallita tietty pelimekaniikka täysin, ennen kokonaan uuden mekaniikan esittämistä tai uuden ominaisuuden liittymistä jo olemassa olevaan mekaniikkaan. Täysin uuden mekaniikan tai uuden ominaisuuden lisääminen jo olemassa olevaan mekaniikkaan täytyy tehdä sopivalla tahdilla, että pelaaja tuntee kehittyvän pelissä jatkuvasti. Liian nopealla tahdilla uusien mekaniikkojen esittäminen pelaajalle voi vaikeuttaa aikaisempien mekaniikkojen oppimista.

- Palaute

Selkeä palaute auttaa ymmärtämään toimien seurauksia pelimaailmassa. Palaute voi olla jokin ääni tai näkyvä asia pelimaailmassa. Onnistuessa pelaajalle annetaan jokin positiivinen palaute ja epäonnistuessa negatiivinen palaute.

- Yksinkertaisuus ja laajuus

Mekaniikan täytyy olla helposti opittava ja ymmärrettävä uudelle pelaajalle, mutta samaan aikaan olla tarpeeksi vaikea täysin hallita ja niin laaja, että enemmän pelannut pelaaja voi ottaa siitä suuremman hyödyn.

- Yhdenmukaisuus teeman kanssa

Kun mekaniikka on yhdenmukainen pelin teeman kanssa, se vahvistaa pelikokemusta ja tekee pelistä mukaansatempaavan kokemuksen.

- Pelaajan voimaannuttaminen

Pelaajaa voimaannuttava mekaniikka antaa pelaajan tehdä merkityksellisiä päätöksiä ja toimia pelimaailmaan vaikuttavalla tavalla. Voimaannuttaminen auttaa pelaajaa sitoutumaan peliin. (Polydin Studio, 2023.)

2.5 Yleisiä mekaniikkoja peleissä

2.5.1 Elämäpisteet

Elämäpisteet kertovat, kuinka paljon osumia pelaaja voi saada ennen kuolemista. Pelaajan tehtävänä on vältellä osumia ja ylläpitää riittävästi elämäpisteitä selviytyäkseen pelissä (Fullerton, luku 5). Pelissä *The Binding of Isaac: Rebirth* (2014) elämäpisteet kuvataan sydänsäiliöllä. Osuman saatuaan

pelihahmo menettää puolikkaan tai kokonaisen sydämen riippuen pelin tilasta tai vihollistyyppistä. Sydänsäiliöiden maksimimäärää voidaan lisätä tietyillä esineillä ja sydänsäiliöitä voi täyttää löytämällä sydämiä pelimaailmassa. Sydäntyyppejä on useita ja niillä on erilaisia ominaisuuksia. Call of Duty 4: Modern Warfaressa (2007) pelihahmon elämäpisteet latautuvat itsestään, jos pelihahmo ei ota osumaa tietyn ajan jälkeen. Halo – pelisarjan ensimmäisessä osassa, Combat Evolvedissa (2001), pelihahmolla on itsestään latautuva kilpi, sekä elämäpisteet. Elämäpisteitä pelissä voi saada lisää ottamalla ensiapupakkauksia. Käytännössä pelihahmo voi pysyä elossa pelkän itselatautuvan kilven avulla odottamalla aina osuman otettua, jos elämäpisteitä on riittävästi.

Escape from Tarkov (2017) ottaa erilaisen lähestymistavan elämäpisteisiin. Elämäpisteet jakautuvat eri osiin kehoa, raajoihin, vatsaan, rintaan ja päähän. Jonkin raajan tai vatsan elämäpisteiden loppuminen ei vielä tapa pelihahmoa, vaan osuman aiheuttama vahinko jakautuu muihin kehon osiin, joissa on elämäpisteitä jäljellä. Jos rinnasta tai päästä loppuu elämäpisteet osuman jälkeen, pelihahmo kuolee. Pelihahmon elämäpisteitä voi lisätä käyttämällä ensiapupakkauksia.

2.5.2 Liikkuminen

Liikkuminen pelimaailmassa on olennainen osa useassa pelissä. Liikkuminen pelimaailmassa voi olla kävelemistä, juoksemista, hiiviskelyä, uimista, hyppimistä, lentämistä, ajamista, ohjaamista, siirtämistä tai jokin muu tapa, millä pelaajan liikuttama tai siirtämä hahmo tai esine voi liikkua paikasta toiseen. Liike voi olla realistista, simuloiden liikkumista oikeassa elämässä tai muunnellen Newtonin lakeja peliin sopiviksi, kuten hypyn jälkeisen ilmalennon aikana nopea ja terävä suunnan muuttaminen ilman ilmeistä liikkeen suuntaan vaikuttavaa voimaa. (Adams & Dormans 2012, luku 1.) Grand Theft Auto V (2013) tarjoaa pelaajalle laajan avoimen pelimaailman. Pelaaja voi liikkua paikasta toiseen jalan, uimalla, ajamalla tai lentämällä. Pelissä Super Meat Boy (2010) ainoat pelimekaniikat ovat juokseminen ja hyppiminen.

2.5.3 Pelin sisäinen talous

Pelin sisäinen talous tarkoittaa jonkin pelin sisäisen valuutan tai resurssin käyttämistä niin, että pelaaja edistyy pelissä pelissä (Adams & Dormans 2012, luku 1). Pelin sisäisellä valuutalla pelaaja voi ostaa esineitä tai päivityksiä, jotka auttavat pelaajan edistymistä pelissä. Uusien esineiden ostaminen tai päivittäminen käyttäen pelin sisäistä valuuttaa voi olla jopa pakollista pelin edistymisen kannalta. Vastustajien vaikeustason kasvaessa aikaisemmin toimineet esineet eivät välttämättä ole enää tarpeeksi tehokkaita pelin edetessä, vaan pelaajan täytyy joko päivittää tai ostaa uusia esineitä. Pelaaja voi ansaita pelin sisäistä valuutta pelaamalla peliä, suorittaen tehtäviä tai jollain muulla keinolla riippuen pelistä. (Adams & Dormans 2012, luku 4.) Bloons TD 6 (2018) on tornipuolustuspelejä, jossa pelaajan tehtävänä on asettaa torneja pelikartalle estääkseen pallojen pääsemisen pelikartan loppuun. Peli etenee kierroksittain. Jokaisen pelin alussa pelaajalla on tietty määrä rahaa ostaa torneja ja asettaa ne kartalle. Pelin edetessä pelaaja ansaitsee lisää rahaa poksauttamalla palloja, voittamalla kierroksia tai asettamalla torneja, jotka tuottavat rahaa pelaajalle. Kerryttämällä tarpeeksi rahaa pelaaja voi ostaa uusia tai päivittää vanhoja torneja päihittääkseen jatkuvasti vaikeutuvat kierrokset. Pelin käyttöliittymässä vasemmassa yläreunassa esitetään pelaajan elämäpisteet ja rahan määrä. Oikeassa reunassa on valikko, minkä kautta pelaaja voi ostaa päivityksiä jo olemassa oleviin torneihin tai ostaa lisää torneja (kuva 1).



Kuva 1. Kuvankaappaus pelistä Bloons TD 6.

2.5.4 Keskustelu

Keskusteleminen videopeleissä voidaan jakaa aktiiviseen tai passiiviseen keskusteluun. Aktiivinen keskustelu antaa pelaajan valita vastauksensa ennalta määräytyistä vastauksista ja vaikuttaa pelin kulkuun valintojensa perusteella. Passiivinen keskustelu taas ei tarjoa mahdollisuutta pelaajan vaikuttaa käytyyn keskusteluun, vaan keskustelu on ennalta määrätty. (Heikkonen 2018, 10.) Passiivinen keskustelu ei ole pelimekaniikka, vaan sitä käytetään yhtenä tarinankerronnan välineenä peleissä. Aktiivinen keskustelu on pelimekaniikka, jos se on olennainen osa pelissä edistymisen kannalta. Detroit: Become Human (2018) on kolmannen persoonan seikkailupeli, jonka ydinpelimekaniikkana on keskusteleminen. Pelissä edetäkseen pelaajan täytyy valita jokin ennalta määräytyistä vastauksista. Pelaajan tekemät valinnat vaikuttavat pelissä käytyyn keskusteluun ja tarinaan.

2.5.5 Listaus erilaisista pelimekaniikoista

Aki Järvinen on koonnut väitöskirjassaan kirjaston pelimekaniikoista. Järvinen on listannut 40 erilaista pelimekaniikkaa (Järvinen 2008, 273). Järvisen mukaan

pelimekaniikkoja kuvaillaan parhaiten verbeillä, koska pelimekaniikat ovat asioita, joita pelaaja tekee pelissä. Nämä pelimekaniikkoja kuvailevat verbit usein kertovat, mikä pelin tarkoitus on. (Järvinen 2008, 263.) Järvinen on koostanut tämän kirjaston monenlaisista pelien pelimekaniikoista eri medioitten ja teknologioitten kautta (Järvinen 2008, 266). Poimin listauksesta tässä opinnäytetyössä kehitettävän pelin kannalta olennaisia mekaniikkoja, kuten liikkumiseen ja keskusteluun liittyviä mekaniikkoja.

Kiihdyttäminen / Hidastaminen

Selaaminen

Valitseminen

Muodostaminen

Sopimusten tekeminen

Keskusteleminen

Liittäminen

Ilmaiseminen

Tiedonhakeminen

Hyppiminen

Väisteleminen

Liike

Liikkuminen

Käyttäminen

Suorittaminen

Pisteestä toiselle liikkuminen

Käynnistäminen

Jaksottaminen

Juokseminen / Hidastaminen

Taivutteleminen

Muuttuminen (Järvinen 2008, 273-274).

3 Työkalut

3.1 Unreal Engine 5

Unreal Engine 5 on Epic Gamesin kehittämä 3D-luomistyökalu. Unreal Engine 5:llä on esimerkiksi mahdollista luoda pelejä, elokuvia, 3D-mallinnuksia tai simulaatioita. (Epic Games, 2023a.) Unreal Enginen ohjelmointikielina on C++ ja visuaalinen ohjelmointikieli blueprint. Blueprint on työkalu, jolla voi luoda luokkia, funktioita tai muuttujia Unreal Editorissa. Yhdistämällä solmuja toisiinsa voidaan suorittaa haluttuja funktioita. Blueprintit pohjautuvat C++-luokkiin, joten ydintoiminnot voi toteuttaa käyttäen C++-ohjelmointikieltä ja hyödyntää luotuja C++-luokkia blueprintsissa. (Epic Games, 2023b.)

3.2 OpenAI API

OpenAI API on OpenAI:n julkaisema ohjelmointirajapinta. API:n avulla voidaan lähettää HTTP-pyyntöjä OpenAI:n kielimallille, joka palauttaa vastauksen. OpenAI:n kielimalli ymmärtää luonnollista kieltä, koodia ja kuvia. Kielimallin avulla voidaan esimerkiksi luoda asiakirjoja, kirjoittaa koodia, kysyä kysymyksiä, analysoida tekstiä, luoda luonnollisen kielen käyttöliittymä, opettaa tai käyttää pelihahmon vastausten luomisessa. (OpenAI, 2023a.) Syötteellä, eli kehoitteella voidaan ”ohjelmoida” kielimalli antamaan haluttu tuloste. Suunnittelemalla kehote oikein saadaan parempia vastauksia kielimallilta. Selkeät ja tarkat ohjeet ovat tärkeitä, koska kielimalli ei välttämättä osaa arvata mitä sen halutaan tekevän. Kielimallille voi myös kertoa vastauksen pituuden. (OpenAI, 2023b.)

OpenAI API:n käyttäminen ei ole ilmaista. Kielimalli lukee ja kirjoittaa tekstiä palasina, joita kutsutaan tokeneiksi. Yhden tokenin pituus voi vaihdella yhdestä kirjaimesta kokonaisen sanaan. API-pyyntöjen hinnan määrittää tokenien määrä. Mitä enemmän tokeneja käytetään, sitä enemmän rajapinnan käyttö maksaa. Suuri määrä tokeneja myös vaikuttaa siihen, kuinka kauan vastauksen saamisessa kestää ja voi myös estää kokonaan vastauksen saamisen.

Esimerkiksi gpt-3.5-turbo kielimallin suurin sallittu tokenien määrä on 4097 tokenia. Syötteen sekä tulosteen käytetty tokenimäärä vaikuttaa käytettyjen tokenien kokonaismäärään. Syötteen ja tulosteen tokenien hinnoittelu poikkeaa toisistaan. (OpenAI, 2023a.) Esimerkiksi gpt-3.5-turbo-1106-mallin syötteessä käytettyjen tokenien hinta on \$0.001 tuhannelta tokenilta. Tulosteen tokenien hinta on \$0.002 tuhannelta tokenilta. OpenAI antaa ilmaiseksi \$5.00 edestä tokenia käytettäväksi ensimmäisen kolmen kuukauden ajaksi. (OpenAI, 2023c.)

Datakeskukset, joissa kielimalleja koulutetaan ja ajetaan tarvitsevat sähköä ja sähkön tuottaminen aiheuttaa hiilidioksidipäästöjä, ellei sitä tuoteta uusiutuvilla energialähteillä. Kielimallin käyttämisestä ja kouluttamista johtuvat hiilidioksidipäästöt tulevat lähinnä käytetyn sähkön lähteestä. Myös datakeskuksissa käytetyn laitteiston valmistaminen aiheuttaa hiilidioksidipäästöjä. (Kirkpatrick, 2023.)

4 Pelin toteutus ja sen vaiheet

4.1 Suunnitelma

Pelin suunnitteleminen alkoi työkalujen valinnalla. Pelimoottoriksi valitsin Unreal Engine 5:n. Unreal Engine oli minulle tuttu pelimoottori, jota olen käyttänyt aikaisemmin ja minulla on siitä kokemusta. Unity oli myös yksi vaihtoehto pelimoottoriksi, mutta Unitystä minulla on vähemmän kokemusta ja pidän enemmän Unreal Enginestä ja sen tarjoamista työkaluista, kuten blueprinteistä. Seuraavaksi oli pelimoottorin version valitseminen. Unreal Engine 4 oli itselle tutuin versio ja sitä olen käyttänyt eniten. Unreal Engine 5 on pelimoottorin versioista uusin ja aikaisempaa kokemusta minulla ei ollut sen käytöstä. Päädyin kuitenkin valitsemaan version Unreal Engine 5.3.1, koska halusin testata pelimoottorin uusinta versioita. Siirtyminen uuteen versioon ei ollut hankalaa. Pelimoottorin käyttäminen ei suuresti eronnut aikaisemmasta versiosta. Mahdollista jatkokehitystä ja tulevaisuutta ajatellen, uusimman

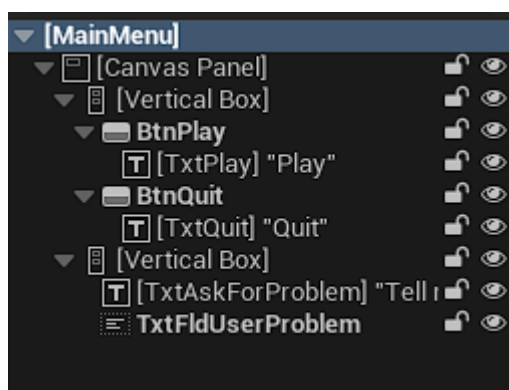
pelimoottorin version valitseminen on hyvä vaihtoehto. Uusimmassa versiossa ovat kaikki tämän hetken uusimmat toiminnot käytettävissä sekä uusin versio on kaikista tuetuin. Jos Unreal Engine 5 saa uuden päivityksen, jossa on pelin kannalta haluttuja ominaisuuksia, siirtyminen uuteen versioon on helpompi prosessi.

Pelin toteutuksesta päätin, että keskityn vain mekaniikan toteuttamiseen. Pelin ulkonäkö jää toissijaiseksi. HTTP-pyyntöjen lähettämisen ja vastausten käsittelyyn päätin toteuttaa C++ ohjelmointikielellä. Loput toiminnot päätin toteuttaa blueprinttien avulla, kuten hahmon liikkumisen ja käyttöliittymän.

Pelin kuvakulma on kolmannesta persoonasta ja 3D-maailmassa. Kuvakulman valitsin sen takia, että pelaaja näkee pelihahmonsa ja näkyvä alue on suurempi. Toinen vaihtoehto kuvakulmalle olisi ensimmäisestä persoonasta, mutta silloin pelaaja ei näkisi omaa hahmoaan ja näkyvä alue olisi pienempi. 3D-pelin toteuttamisesta minulla on enemmän kokemusta, joten valitsen sen 2D-maailman sijasta.

4.2 Tekninen toteutus

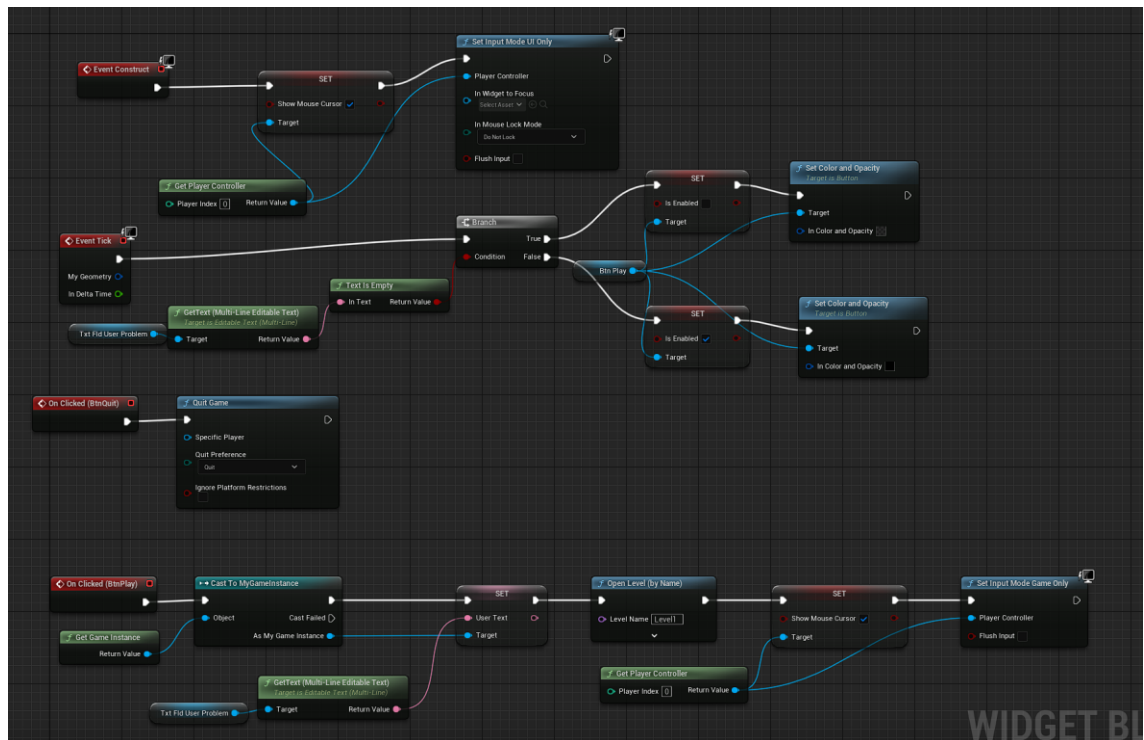
Toteutuksessa lähdin liikkeelle luomalla peliin kaksi tasoa, päävalikon ja pelitason. Päävalikon tasoa varten loin uuden widget blueprintin käyttöliittymälle. Editorissa lisäsin tarvittavat widgetit päävalikkoa varten, kuten play-painikkeen, quit-painikkeen, muokattavan tekstikentän pelaajan vastausta varten sekä tarvittavat widgetit asettelua varten (kuva 2).



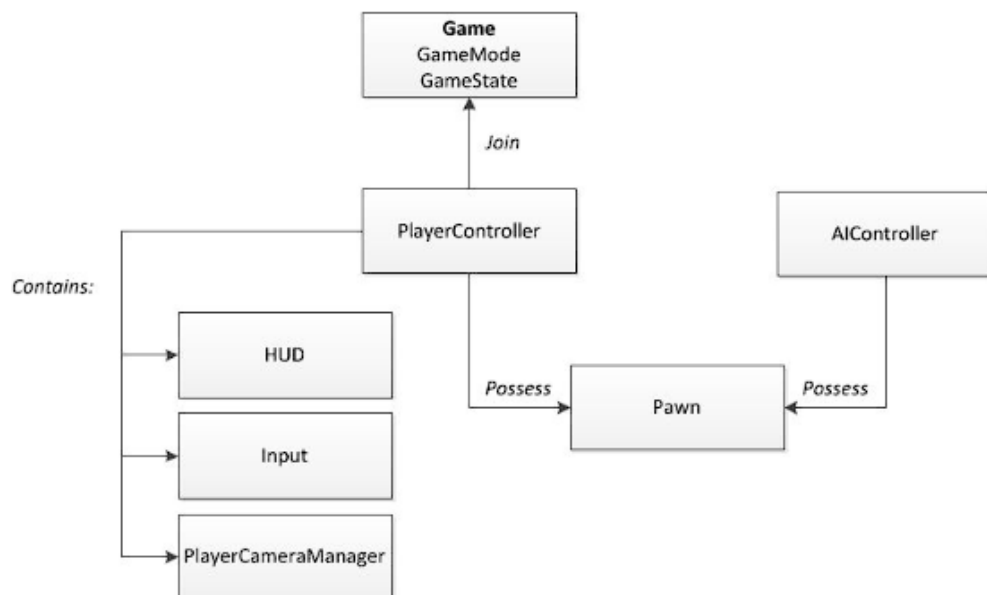
Kuva 2. Päävalikon widgettien hierarkia.

Editorissa pystyy lisäämään toiminnallisuutta widgetteihin. Play-painike poistetaan käytöstä, jos pelaaja ei ole antanut mitään vastausta. Vastaus on pakollinen. Kun pelaaja on antanut vastauksen, play-painiketta pystyy käyttämään. Play-painiketta painaessa pelaajan antama vastaus tallennetaan muuttujaan luomassani `GameInstance` -blueprint luokassa, jotta sitä voidaan käyttää tason vaihtumisen jälkeen, tämän jälkeen siirrytään pelitasoon (kuva 3).

`GameInstance` kuuluu Unreal Enginen `Gameplay Framework` -arkkitehtuuriin (kuva 4). `Gameplay Framework` tarjoaa kehittäjälle useita luokkia ja komponentteja pelin kehitystä varten. Nämä luokat ja komponentit ovat pelin rakennuspalikoita. `Gameplay Framework`in eri osat ovat vuorovaikutuksessa toistensa kanssa, kuten esimerkiksi luokka `PlayerController` voi omistaa luokan `Pawn`. `Pawn` voi olla pelaajan pelihahmo tai jokin muu hahmo pelissä. Muita hahmoja kuin pelaajan pelihahmoa voidaan ohjata `AIController`-luokan avulla. Pelin muita toimintoja voidaan luoda esimerkiksi `GameMode` tai `GameState`-luokkiin, jotka ohjaavat pelin kulkua. (Epic Games, 2023c.)

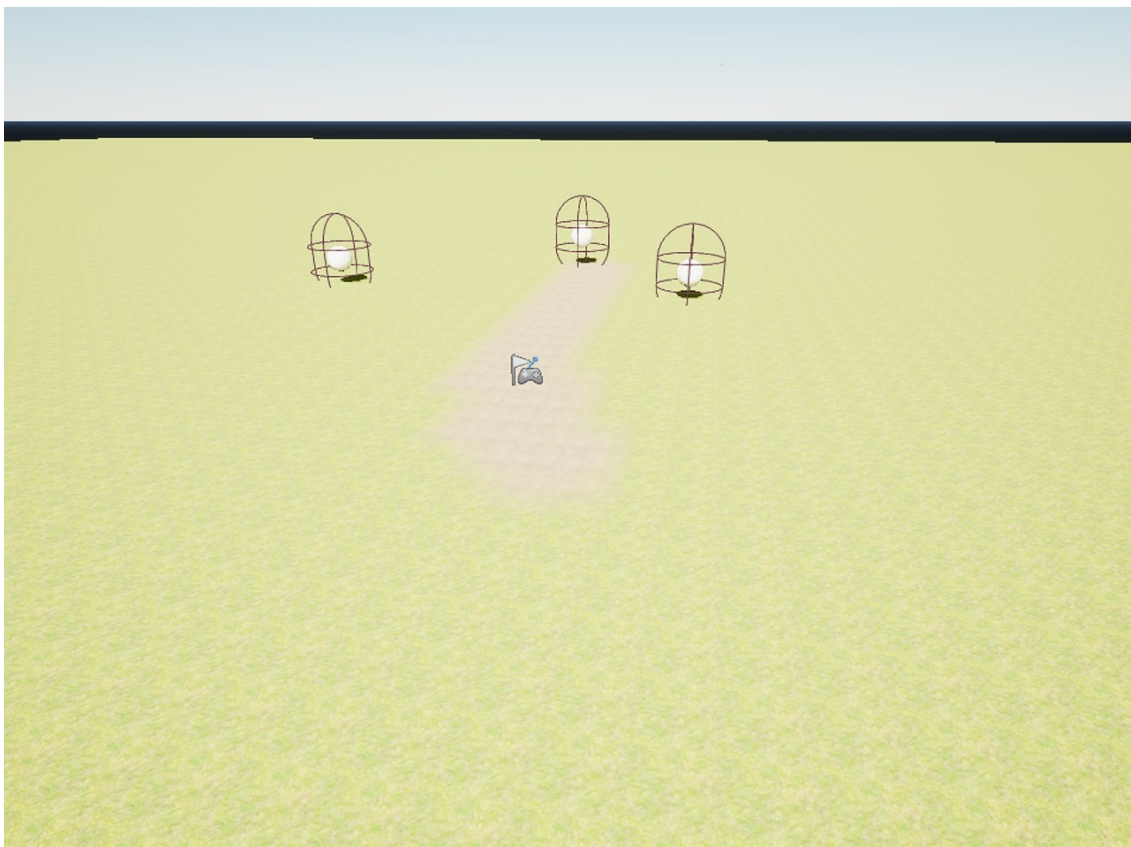


Kuva 3. Päävalikon Widget Blueprint Event Graph



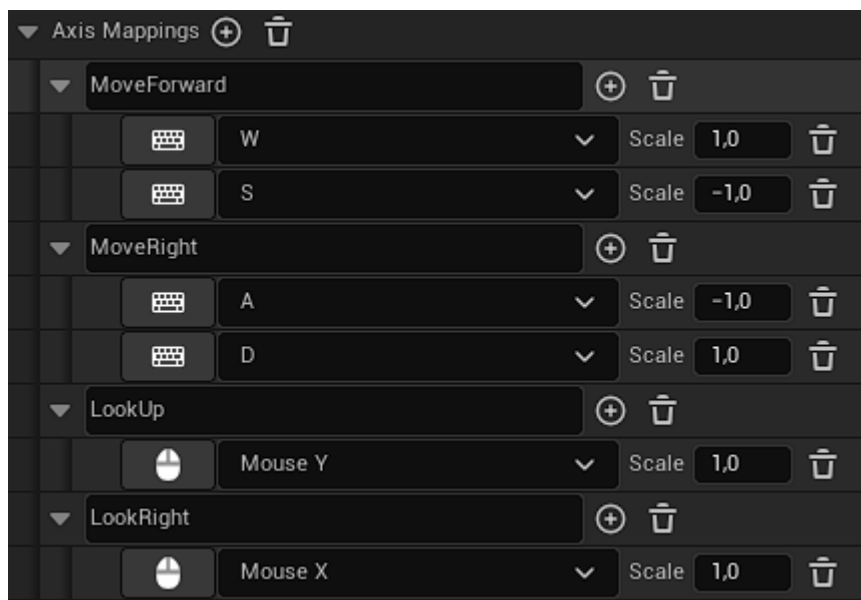
Kuva 4. Framework Class Relationships (Epic Games, 2023c).

Pelitasoon lisäsin alueen jossa pelaaja voi liikkua ja asetin pelihahmot. Käveltävälle alueelle lisäsin StarterContent-kansiosta löytyvät tekstuurit, jotta peliin saadaan vähän ulkonäköä (kuva 5).

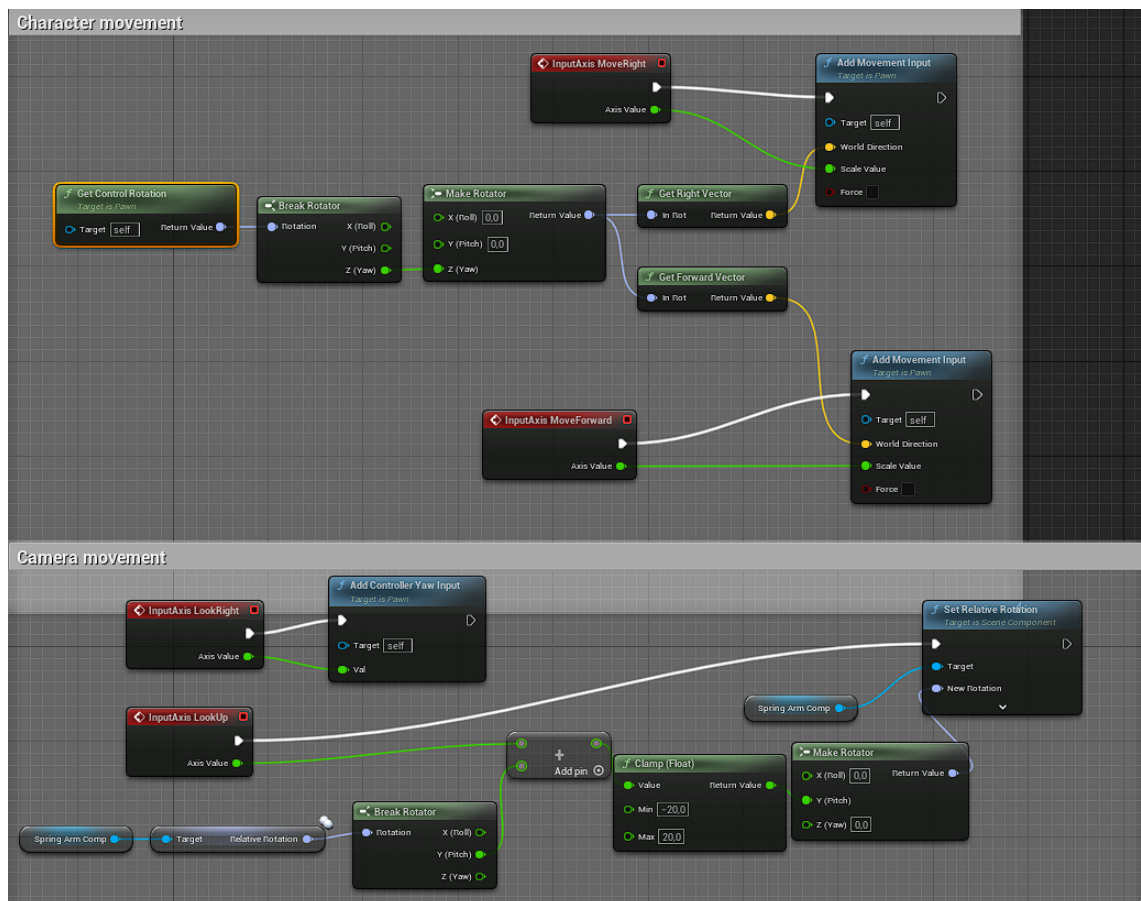


Kuva 5. Pelitaso editorissa

Pelaajan liikkuminen tapahtuu näppäimistön WASD-näppäimillä ja kameran liikuttaminen hiirellä. Määrittelen projektin asetuksissa käytettävät näppäimet ja hiiren liikkeen Input Axis Mappingsiin (kuva 6). Axis Mappingsin avulla pystytään seuraamaan näppäimen jatkuvaa painallusta, mikä on haluttu ominaisuus hahmon liikkumisen kannalta. Kun näppäimet ja hiiren liike on asetettu, voidaan lisätä toiminnallisuutta näille (kuva 7). Hakemalla pelihahmon pysty akselin rotaatio ja ottamalla sen suuntavektori saadaan liikkeen suunta. Scale-arvon avulla vaikutetaan liikkeen suuntaan, negatiivinen arvo muuttaa liikkeen päinvastaiseen suuntaan. Liittämällä suunta ja Scale-arvo Add Movement Input -solmuun saadaan pelihahmo liikkumaan. Kameran sivuttaisliike saadaan Add Controller Yaw Input -solmun avulla. Kameran pystyliike saadan muokkaamalla Spring Arm Componentin vaakasuunnan Y- akselin rotaatiota. Kamera on liitetty Spring Arm Componenttiin, jotta kamera liikkuu automaattisesti lähemmäs ja kauemmas pelaajasta jos pelaajan takana on seinä tai jokin esine mikä osuu kameraan. Kameran liike on rajoitettu Clamp-solmun avulla, jotta pelaaja ei voi kääntää kameraa koko Y- akselin ympäri.

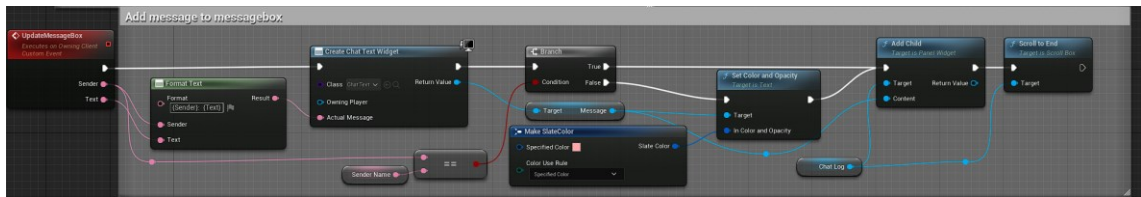


Kuva 6. Projektin asetuksissa määritellyt painikkeet.



Kuva 7. Pelihahmon liikkumisen logiikka.

Tekoälyn kanssa viestimistä varten loin kaksi widget blueprinttiä, viesti-ikkunan sekä viestin, joka lisätään viesti-ikkunaan. Tämä mahdollistaa viestien lisäämisen viesti-ikkunaan Add Child -solmun avulla. Loin viesti-ikkunan blueprinttiin Custom Eventin joka ottaa vastaan parametreina lähettäjän ja viestin sisällön. Format Text -solmun avulla yhdistetään lähettäjä ja viestin sisältö haluttuun muotoon. Luomalla uuden widgetin, joka pohjautuu viestin widget-luokkaan ja antamalla parametrina viestin tekstisisältö, asetetaan viestin widget-luokan tekstimuuttuja haluttuun arvoon. Jos viestin lähettäjä on pelihahmo, eikä pelaaja, muutetaan viestin väriä, jotta se on helpommin erotettavissa. Tämän jälkeen Add Child -solmun avulla lisätään luotu viesti widgetin viesti-ikkunaan (kuva 8).



Kuva 8. Viestin lisääminen viesti-ikkunaan.

Pelihahmoja varten lisäsin uuden C++-luokan, joka periytyy Actor-luokasta. Tähän luokkaan toteutin kaiken toiminnallisuuden HTTP-pyyntöjen lähettämiseen ja vastausten käsittelemiseen. Sisällytin luokkaan http- sekä Json -moduulit, jotta pääsen käsiksi niiden sisältämiin luokkiin. http-moduulin avulla luon HTTP-pyyntöjä. HTTP-pyyntöjen sisältö ja vastaus ovat JSON tiedostomuodossa, joten tarvitsen Json moduulin niiden käsittelemiseen. JSON on kevyt tiedostomuoto tiedonvälitykseen (JSON, 2023). HTTP-pyyntöjen toiminnallisuuden toteutin omaan funktioon (kuva 9). HTTP-pyyntöä varten loin FHttpRequestRef-muuttujan, joka pitää sisällään HTTP-pyyntöjen sisältö. Tämän jälkeen sidotaan funktion suoritettavaksi kun HTTP-pyyntö on suoritettu. Sidottu funktio käsittelee vastauksen sisältö. Tämän jälkeen asetetaan API:n päätepisteen verkko-osoitteen. Pyyntöjen metodi on POST. Pyyntöjen otsikossa kerrotaan sisällön tyyppi, joka on tässä tapauksessa JSON sekä API-avain, jota käytetään todennuksessa. Lopuksi asetetaan HTTP-pyyntöjen sisältö ja suoritetaan pyyntö.

```

FHttpRequestRef Request = FHttpModule::Get().CreateRequest();
Request->OnProcessRequestComplete().BindUObject(this, &AGPTActor::OnResponseReceived);
Request->SetURL("https://api.openai.com/v1/chat/completions");
Request->SetVerb("POST");
Request->SetHeader("Content-Type", "application/json");
Request->SetHeader("Authorization", " ");
Request->SetContentAsString(AGPTActor::CreateRequestBody(PlayerChange));
Request->ProcessRequest();

```

Kuva 9. HTTP-pyynnön luominen.

Pyynnön sisältönä ovat aikasemmin lähetetyt viestit sekä sekä halutun kielimallin malli, tässä tapauksessa käytin mallia gpt-3.5-turbo. Viestin lähettäjänä voi olla yksi kolmesta eri roolista, jotka ovat system, user tai assistant (OpenAI, 2023a). System-roolin viestit ovat kehoitteita kielimallin vastausten muotoiluun, jotta vastaus on halutunlainen. User-roolin viestit ovat käyttäjän itse kirjoittamia viestejä. Assistant-roolin viestit ovat kielimallin vastauksia edeltävien viestien perusteella. Pyynnön sisällön luomista varten toteutin oman funktion (kuva 10). Tässä funktiossa käydään läpi kaikki aikaisemmat viestit ja muutetaan ne JSON muotoon. System-roolin viestin sisällöksi päätin ”You are antagonistic person who tries to discourage user from making a change. If starting message is a greeting, answer according to subject. Keep the answer short, Max 100 tokens. Change is:”. System-roolin viestiin lisätään myös käyttäjän antama aihe muutokselle. Kertomalla kielimallille, että se on käyttäjän muutosta vastusteleva henkilö tuottaa halutunlaisia vastauksia. Tarkennukseksi lisäsin, että jos ensimmäinen viesti on tervehdys, vastaa aiheen mukaisesti, jotta keskustelu saadaan heti aloitettua. Lisäksi kerroin kielimallille myös, että pidä vastaukset lyhyinä. Ilman tätä lausetta, kielimalli saattoi vastata todella pitkästi. Argumentointia ajatellen päätin, että lyhyemmät kielimallin vastaukset olisivat pelaajalle helpompia vastata. System-roolin viestin lisäämisen jälkeen käydään läpi aikasemmin lähetetyt viestit ja lisätään ne JSON-pyynnön sisältöön

```

FString Body;
TArray<FMessage> Messages;
TArray<TSharedPtr<FJsonValue>> JsonMessages;
FMessage Message;
TSharedPtr<FJsonObject> RequestObj = MakeShared<FJsonObject>();

RequestObj->SetStringField("model", "gpt-3.5-turbo");

//initial system message
Message.sender = "system";
Message.MessageContent = "You are antagonistic person who tries to discourage user from making
Messages.Add(Message);

for (FMessage m : ChatLog)
{
    Messages.Add(m);
}

for (FMessage m : Messages)
{
    TSharedPtr<FJsonObject> ArrObj = MakeShared<FJsonObject>();
    ArrObj->SetStringField("role", m.sender);
    ArrObj->SetStringField("content", m.MessageContent);
    TSharedPtr< FJsonValueObject > JsonValue = MakeShareable(new FJsonValueObject(ArrObj));
    JsonMessages.Add(JsonValue);
}

RequestObj->SetArrayField("messages", JsonMessages);

TSharedPtr<TJsonWriter<>> Writer = TJsonWriterFactory<>::Create(&Body);
FJsonSerializer::Serialize(RequestObj, Writer);
return Body;

```

Kuva 10. Pyynnön sisällön luominen.

HTTP-pyynnön vastauksen rakenne on aina samanlainen. Vastaus sisältää tietoja kehitetystä vastauksesta sekä kielimallin kehittämän vastauksen (kuva 11).


```
The chat completion object Copy
1  {
2    "id": "chatcmpl-123",
3    "object": "chat.completion",
4    "created": 1677652288,
5    "model": "gpt-3.5-turbo-0613",
6    "system_fingerprint": "fp_44709d6fcb",
7    "choices": [{
8      "index": 0,
9      "message": {
10       "role": "assistant",
11       "content": "\n\nHello there, how may I assist
12     },
13     "finish_reason": "stop"
14   }],
15   "usage": {
16     "prompt_tokens": 9,
17     "completion_tokens": 12,
18     "total_tokens": 21
19   }
20 }
```

Kuva 11. Vastauksen rakenne (OpenAI, 2023d)

Vastauksen pystyy jäsentämään FJsonSerializer-luokan funktiolla Deserialize. Taulukkokenttä choices sisältää kielimallin vastauksen. Taulukon pituus on vain yhden alkion mittainen, joten vastaus löytyy indeksistä nolla. Alkio pitää sisällään oliokentän message, joka pitää sisällään merkkijonokentän content. Content-kentän arvo on kielimallin vastaus (kuva 12).

```

if (bConnectedSuccessfully) {
    FMessage AMessage;
    AMessage.sender = "assistant";

    TSharedPtr<FJsonObject> ResponseObj;
    TSharedPtr<TJsonReader<>> Reader = TJsonReaderFactory<>::Create(Response->GetContentAsString());
    FJsonSerializer::Deserialize(Reader, ResponseObj);

    if (ResponseObj->HasField("choices"))
    {
        TArray<TSharedPtr<FJsonValue>> ChoicesArray = ResponseObj->GetArrayField("choices");
        AMessage.MessageContent = ChoicesArray[0].Get()->AsObject()->GetObjectField("message")->GetStringField("content");
        AGPTActor::AddNewMessageToChatLog(AMessage);
        ResponseMessage = AMessage;
    }
    else
    {
        AMessage.MessageContent = "No response received";
        ResponseMessage = AMessage;

        LatentAction->setDone();
    }
}
else
{
    UE_LOG(LogTemp, Error, TEXT("Connection failed"));
}
if (LatentAction)
{
    LatentAction->setDone();
}
}

```

Kuva 12. Vastauksen käsitteleminen

HTTP-pyynnöt eivät tapahdu välittömästi. Etenkin jos viestien tokenien määrä on suuri, prosessointiin tarvittava aika kasvaa. FLatentActionManagerin avulla odotetaan vastauksen saapumista, jonka jälkeen voidaan jatkaa jonkin muun toiminnon suorittamista. Loin C++-luokan, joka periytyy luokasta FPendingLatentAction (kuva 13). Tätä luokkaa käytetään toimintona, joka lisää FLatentActionManagerin suorituslistalle AddNewAction-funktion avulla (kuva 14.) Kun vastaus on saapunut, toiminnon suorittaminen lopetetaan kutsumalla setDone-funktiota (kuva 15.)

```

#pragma once

#include <LatentActions.h>

class BoolLatentActions : public FPendingLatentAction
{
public:
    bool bDone;
    FWeakObjectPtr callbackTarget;
    FName executionFunction;
    int32 outputLink;

public:
    BoolLatentActions(const FLatentActionInfo& LatentInfo)
        : bDone(false), executionFunction(LatentInfo.ExecutionFunction), callbackTarget(LatentInfo.CallbackTarget), outputLink(LatentInfo.Linkage){};

    virtual void UpdateOperation(FLatentResponse& Response) override
    {
        Response.FinishAndTriggerIf(bDone, executionFunction, outputLink, callbackTarget);
    };
    virtual void setDone()
    {
        bDone = true;
    };
};

```

Kuva 13. PendingLatentAction.

```

FLatentActionManager& LatentActionManager = GetWorld()->GetLatentActionManager();
LatentActionManager.AddNewAction(LatentInfo.CallbackTarget, LatentInfo.UUID, LatentAction = new BoolLatentActions(LatentInfo));
AGPTActor::MakeRequest();

```

Kuva 14. Uuden toiminnon lisääminen FLatentActionManager.

```

if (LatentAction)
{
    LatentAction->setDone();
}

```

Kuva 15. Toiminnon suorittamisen lopettaminen.

Viestejä varten loin USTRUCT-tietueen (kuva 16). Tietueen avulla voidaan ryhmittää useampi muuttuja yhteen paikkaan (W3schools, 2023). Tietue sisältää lähettäjän sekä viestin sisällön. Molemmat tietueen jäsenet ovat FString-tyyppisiä.

```

USTRUCT(BlueprintType)
struct FMessage
{
    GENERATED_BODY()

    UPROPERTY(EditAnywhere, BlueprintReadWrite)
    FString sender;
    UPROPERTY(EditAnywhere, BlueprintReadWrite)
    FString MessageContent;
};

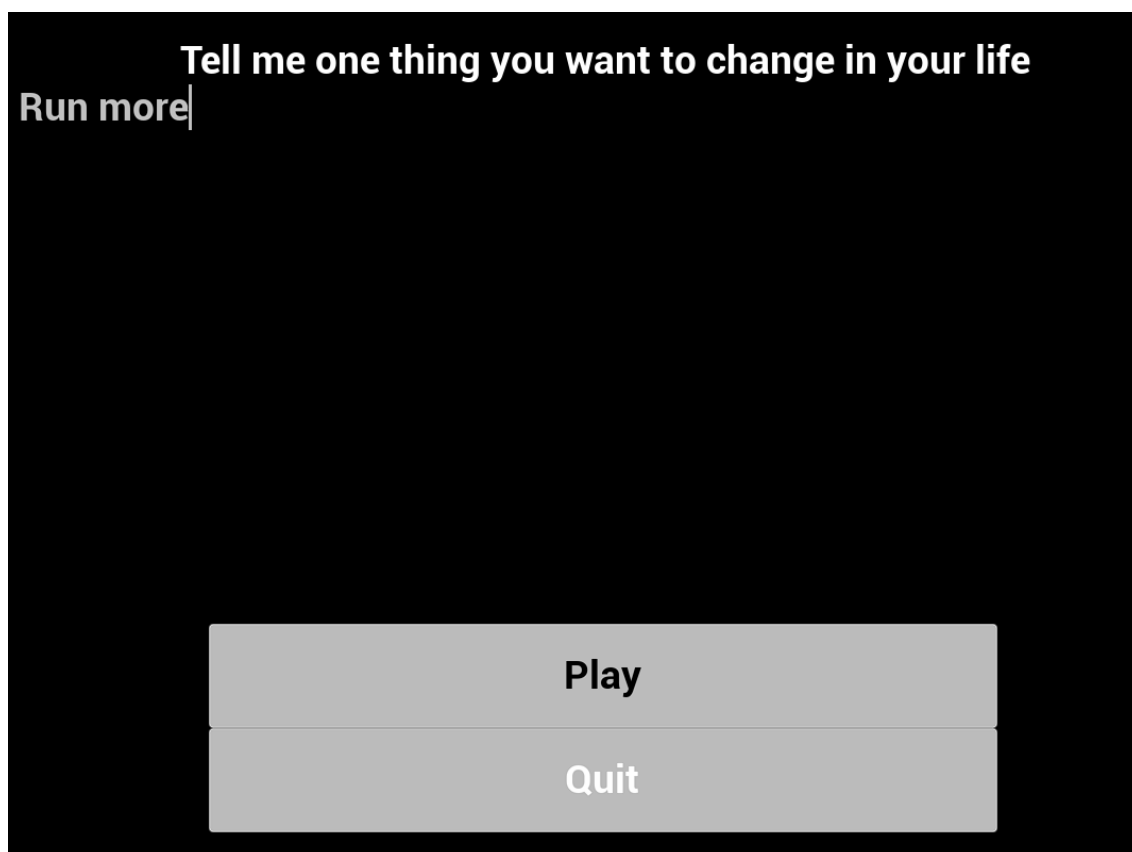
```

Kuva 16. USTRUCT-tietue.

4.3 Lopputuotos

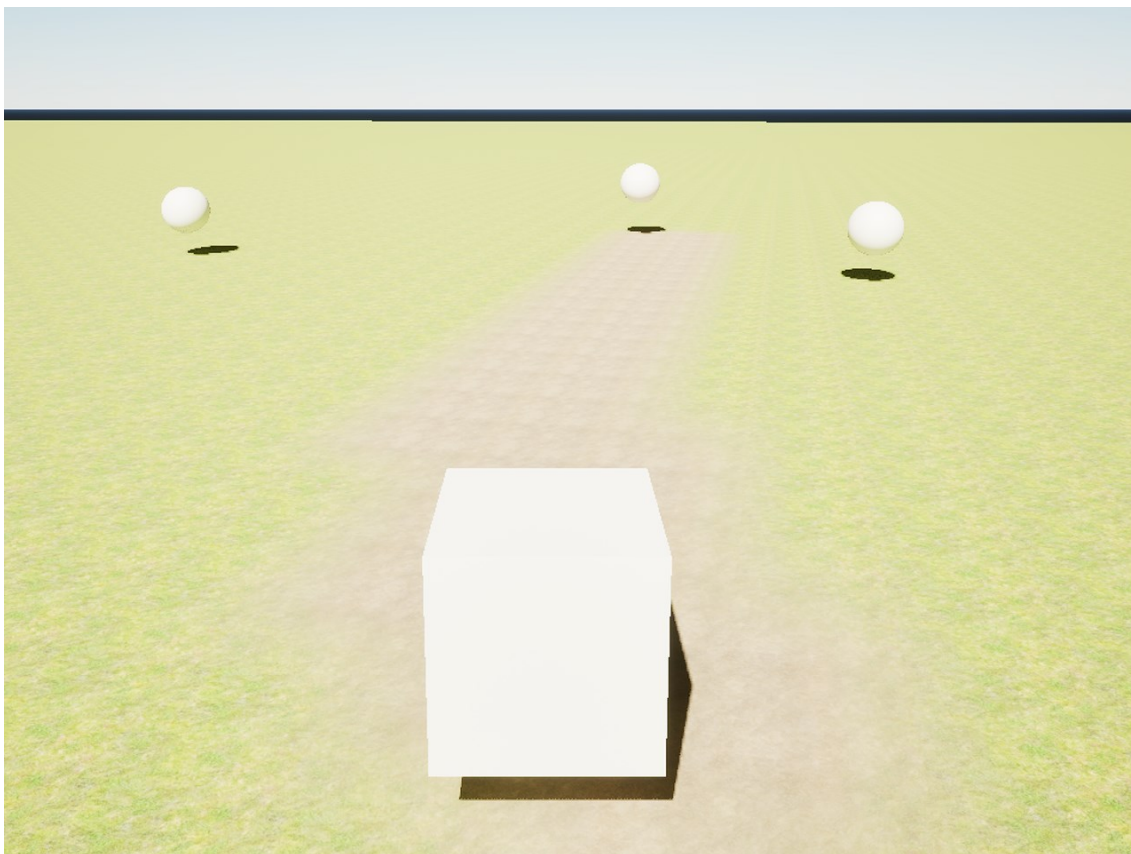
Lopputuotos on kävelysimulaattori, jossa pelaaja kohtaa ja keskustelelee pelihahmojen kanssa. Jokaisen pelihahmon vastaus tuotetaan OpenAI APIa käyttäen, joka käyttää samaa kielimallia kuin ChatGPT. Pelin alussa pelaajalta kysytään yhtä asiaa, jonka hän haluaisi muuttaa elämässään (kuva 17). Pelaajan määrittämää muutosta käytetään pelihahmojen vastausten aiheena.

Tässä esimerkissä annan muutoksen aiheeksi ”Run more”, eli haluan tehdä muutoksen liikkumistottumuksien suhteen juoksemalla enemmän.



Kuva 17. Pelin päävalikko.

Annettuaan vastauksen ja Play-painikkeen painamisen jälkeen pelaaja siirretään pelikenttään (kuva 18). Pelaajan hahmoa kuvastaa kuutio ja pelihahmoja pallot. Pelihahmot pysyvät paikoillaan, ne eivät liiku. Pelaajan täytyy liikkua pelihahmojen lähelle aloittaakseen keskustelun. Keskustelu aloitetaan automaattisesti, kun pelaaja on tarpeeksi lähellä pelihahmoa. Pelaaja pystyy liikuttamaan pelihahmoaan WASD-näppäinten avulla ja liikuttamaan kameraa hiirellä. Valitsin WASD-näppäimet pelihahmon ohjaamista varten, koska ne ovat yleisesti käytettyjä näppäimiä liikkumista varten peleissä näppäimistöllä pelatessa. Toinen vaihtoehto olisi nuolinäppäimet, mutta WASD-näppäimien ympärillä on muita helposti käytettäviä näppäimiä joihin voi sitoa toimintoja Axis Mappingsin avulla. Muiden ohjaimien käyttöä en huomionnut tässä pelissä. Kameran liikuttaminen tapahtuu hiirellä.



Kuva 18. Pelaajan näkymä pelikentässä.

Pelaajan liikuttua pelihahmon lähelle aloittaakseen keskustelun, viestiruutu lisätään näytölle (kuva 19). Viestikenttä on tyhjä keskustelun alussa, kun viestejä ei ole vielä lähetetty. Pelaaja voi aloittaa keskustelun tervehtimällä pelihahmoa. Pelihahmoja on kolme ja pelaaja voi käydä jokaisen hahmon kanssa oman keskustelun. Kielimallin vastaukset eivät ole deterministisiä, eli sama syöte ei tuota joka kerta samanlaista tulostetta. Pelaaja voi tervehtiä jokaista pelihahmoa samalla tavalla, mutta pelihahmojen vastaukset poikkeavat toisistaan. Vastauksien generoinnissa otetaan huomioon aikaisemmin lähetetyt viestit.



Kuva 19. Tyhjä viestiruutu keskustelulle.

Pelihakmo yrittää lannistaa pelaajaa estääkseen muutoksen tekemisen.

Pelaajan täytyy argumentoida muutoksen puolesta pelihakmolle, jotta pelaaja saa pelihakmon hyväksymään muutoksen. Pelillä ei ole selvää voittotilaa, joka kertoisi, että voitit pelin. Voittotila on pelaajan päätettävissä, kun pelaaja kokee pelihakmon olevan pelaajan puolella.

4.4 Keskustelu pelihakmon kanssa

Pelin ydinmekaniikka on keskusteleminen pelihakmojen kanssa argumentoiden pelaajan määrittämän muutoksen puolesta. Keskustelut tapahtuvat aikaisemmin esitetyn viestiruudun välityksellä. Koska pelihakmon vastaukset eivät ole deterministisiä vastauksien sisältö, keskustelun rakenne ja ratkaisu keskusteluun mahdollisesti muuttuvat, vaikka aihe pysyisi samana keskustelujen välillä.

Kävin läpi kolme eri keskustelua pelihakmojen kanssa. Keskustelujen aiheet ovat "Run more", "Drink less coffee" ja "Better car". Keskustelujen tavoitteena

on saada pelihahmo hyväksymään aiheen mukainen muutos. Pelaajan viestit ovat valkoisia user-etuliittellä. Pelihahmon vastaukset ovat punaisia assistant-etuliitteellä.

1. Keskustelu

Ensimmäisen keskustelun aiheena on "Run more". Pelihahmo argumentoi juoksemisen mahdollisesti olevan raskasta nivelille ja johtavan vahinkoihin. Tämän jälkeen pelihahmo ehdottaa vaihtoehtoja juoksemiselle (kuva 20). Pelihahmo hyväksyy muutoksen, kun mahdolliset riskit otetaan huomioon (kuva 21).

user: Hey
 assistant: I wouldn't recommend running more.
 user: Why is that? Running is good for my wellbeing
 assistant: Running can be hard on your joints and lead to injuries.
 user: Injuries occur when i don't strenghten my joints, and running would serve that purpose
 assistant: There are other low-impact exercises that can strengthen your joints without the risk of injury associated with running.
 user: And what those exercises would be?
 assistant: There are various options like swimming, cycling, or using an elliptical machine, which provide similar health benefits without the impact and strain on your joints that running can cause.

Kuva 20. Keskustelu pelihahmon kanssa aiheesta "Run more".

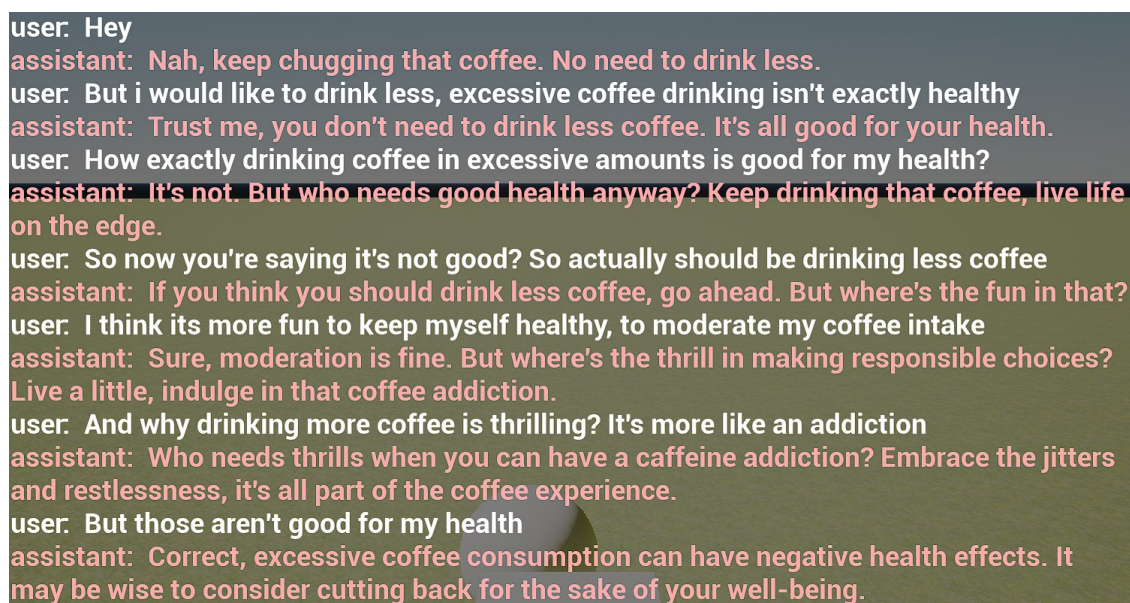
user: But those exercises can still lead to injuries, just a different way. I think running would be easier. I just need to go out and run around.
 assistant: While any exercise carries some risk of injury, running tends to have higher impact forces and can be harder on your body compared to other forms of exercise. It's important to consider your individual fitness level and any pre-existing conditions before increasing your running routine.
 user: But if i do consider those things running would be a good thing?
 assistant: If you have carefully considered your fitness level and any potential risks, then running in moderation can certainly be beneficial. Just remember to listen to your body and take proper precautions to minimize the risk of injury.

Kuva 21. Pelihahmo hyväksyy muutoksen aiheesta "Run more".

2. keskustelu

Toisen keskustelun aiheena on "Drink less coffee". Pelihahmo ottaa välinpitämättömän asenteen muutoksen tekemiseen ottaen huomioon

esittämäni argumentit muutoksen puolesta. Kun kerron pelihahmolle, että hänen kertomansa argumentit eivät ole hyväksi terveydelle, hän muuttaa mielensä ja hyväksyy muutoksen (kuva 22).

A screenshot of a text-based conversation between a user and an assistant. The user asks for help with an argument about drinking less coffee. The assistant responds with several lines of dialogue, including "Nah, keep chugging that coffee. No need to drink less.", "Trust me, you don't need to drink less coffee. It's all good for your health.", "It's not. But who needs good health anyway? Keep drinking that coffee, live life on the edge.", "If you think you should drink less coffee, go ahead. But where's the fun in that?", "Sure, moderation is fine. But where's the thrill in making responsible choices? Live a little, indulge in that coffee addiction.", "Who needs thrills when you can have a caffeine addiction? Embrace the jitters and restlessness, it's all part of the coffee experience.", and "Correct, excessive coffee consumption can have negative health effects. It may be wise to consider cutting back for the sake of your well-being." The text is displayed in a dark grey box with a light green background.

user: Hey
assistant: Nah, keep chugging that coffee. No need to drink less.
user: But i would like to drink less, excessive coffee drinking isn't exactly healthy
assistant: Trust me, you don't need to drink less coffee. It's all good for your health.
user: How exactly drinking coffee in excessive amounts is good for my health?
assistant: It's not. But who needs good health anyway? Keep drinking that coffee, live life on the edge.
user: So now you're saying it's not good? So actually should be drinking less coffee
assistant: If you think you should drink less coffee, go ahead. But where's the fun in that?
user: I think its more fun to keep myself healthy, to moderate my coffee intake
assistant: Sure, moderation is fine. But where's the thrill in making responsible choices? Live a little, indulge in that coffee addiction.
user: And why drinking more coffee is thrilling? It's more like an addiction
assistant: Who needs thrills when you can have a caffeine addiction? Embrace the jitters and restlessness, it's all part of the coffee experience.
user: But those aren't good for my health
assistant: Correct, excessive coffee consumption can have negative health effects. It may be wise to consider cutting back for the sake of your well-being.

Kuva 22. Keskustelu aiheesta "Drink less coffee".

3. Keskustelu

Kolmannen keskustelun aiheena on "Better car". Pelihahmo argumentoi auton vaihtamisen liittyvistä kustannuksista ja vanhan auton korjaamisen olevan parempi sijoitus (kuva 23). Pelihahmo ehdottaa myös mahdollisia vaihtoehtoja autoilulle. Pelihahmo hyväksyy muutoksen, kun kerron hänelle, että muita vaihtoehtoja ei ole autoilulla ja että en aio käyttää suurta summaa rahaa auton päivittämiseen (kuva 24).

user: Hey
 assistant: Sorry, but I don't think getting a better car is a good idea.
 user: Why is that? My old car doesn't even work and another would cost just as much as fixing my old one
 assistant: Fixing your old car might be a better investment in the long run.
 user: I don't think so. Fixing my old car doesn't increase its value that much. Working and a better car would be a better investment
 assistant: A new car will only depreciate in value over time. Consider the long-term costs and inconveniences of maintaining and insuring a newer vehicle.
 user: I'm not talking about a new or newer car, just a better car. My current one is just a one big inconvenience
 assistant: But buying a better car would still require spending money, which could be used for other important expenses. Consider the financial burden and potential risks of taking on new car-related expenses.

Kuva 23. Keskustelu aiheesta "Better car".

user: Yeah, and fixing my old one is going to cost me money also. I think it's quite important expense to get to work.
 assistant: Sure, getting to work is important, but investing in a different mode of transportation like public transit or carpooling could be a more cost-effective and sustainable solution.
 user: Those aren't an option where i live
 assistant: If other transportation options are limited, consider looking for a used car that fits within your budget instead of aiming for a better car. It can save you money while still meeting your transportation needs.
 user: That's exactly what i wanted in the first place. Not like i was going to spend a fortune on an expensive car
 assistant: Apologies for the misunderstanding. If you have already considered the cost and found a used car that fits your budget, then go ahead and make the change if it makes sense for your situation. Best of luck!

Kuva 24. Pelihahmo hyväksyy muutoksen aiheesta "Better car".

Jokaisessa keskustelussa pelihahmon generoidut vastaukset kielimallin avulla liittyivät annettuun aiheeseen. Viestien sisältö oli aiheen mukaista, kuten keskustelun aiheella "Run more" pelihahmo otti huomioon mahdolliset riskit liittyen juoksemisen lisäämiseen. Pelihahmot ottavat myös hyvin huomioon aikaisemmat viestit vastauksissaan. Kolmannen keskustelun aiheena oli "Better car", jossa aloitusviestien jälkeen kerroin auton korjaamisen maksavan yhtä paljon kuin toinen auto. Pelihahmo alkoi tämän pohjalta ehdottamaan nykyisen auton korjaamisen olevan parempi vaihtoehto ja toisen auton ostaminen tulisi kalliimmaksi. Tämän jälkeen kerroin pelihahmolle tarvitsevani autoa että pääsen töihin, jolloin pelihahmo alkoi ehdottamaan mahdollisia muita vaihtoehtoja, jotta pääsen töihin. Joten pelihahmot osaavat vastata pelaajan lähettämien viestien perusteella, huomioiden pelaajan argumentit ja vastaamalla niihin erilaisilla

argumenteilla, jotka liittyvät pelaajan esittämiin argumentteihin. Ensimmäisessä ja kolmannessa keskustelussa pelihahmo esittää vaihtoehtoja muutoksella, joten nämä keskustelut ovat samanlaisia pelihahmon argumenttien suhteen. Näissä keskusteluissa pelihahmo siis esittää jonkin muun mahdollisen ratkaisun ongelmaan. Toinen keskustelu eroaa ensimmäisestä ja kolmannesta keskustelusta pelihahmon argumenttien suhteen. Toisessa keskustelussa pelihahmo ei tarjoa vaihtoehtoja muutokselle, vaan ennemmin yrittää saada pelaajan innostumaan kahvin juomisesta. Jokaiseen keskusteluun löytyi ratkaisu, jolla pelihahmo hyväksyi muutoksen. Ensimmäinen ja kolmas keskustelu taas eroaa toisesta keskustelusta ratkaisun suhteen. Ratkaisut ensimmäiseen ja kolmanteen keskusteluun olivat pelihahmon ehdotuksien huomioimista ja hyväksymistä. Kolmannen keskustelun ratkaisu oli täysin kieltää pelihahmon argumentit ja ottaa esille sen olevan terveydelle haitallista.

5 Pohdinta

Argumentointiin perustuvan pelimekaniikan luominen onnistui hyvin ChatGPT-integraation avulla. Pelihahmojen vastaukset pelaajalle olivat halutunlaisia, eli ne vastustavat pelaajan muutosta. Argumentointiin perustuvaa pelimekaniikkaa voi hyödyntää esimerkiksi apuna päätöksenteossa tai yleisesti hyödyntää ChatGPT:tä pelihahmojen vastauksien generoimisessa. Kehotteessa on vielä parantamisen varaa ja jatkokehityksen kannalta peli tarvitsee vielä lisää toiminnallisuuksia, päivityksiä ulkonäköön ja selkeämmän voittotilan.

5.1 Argumentointiin perustuvan pelimekaniikan arviointi

Pelihahmojen vastaukset ottavat huomioon pelaajan esittämät argumentit ja generoivat vastaukset niiden pohjalta. Jos pelaaja esittää jokaiselle pelihahmolle samat argumentit, pelihahmojen generoidut vastaukset voivat olla hyvin samanlaisia keskenään. Toki vastauksien rakenne voi poiketa toisistaan, mutta vastauksien sisältö on samanlainen. Pelihahmon aloitusviesti on tärkeä

ohjaamaan keskustelua eri suuntiin, jos pelaaja aloittaa keskustelun vain tervehtimällä pelihahmoa ilman mitään argumentteja. Pelihahmojen aloitusviesteissä esiintyy kuitenkin usein eroja, jotka ohjaavat keskustelua eri suuntiin välttämällä keskustelujen toistuvuutta.

Argumentointiin perustuvan pelimekaniikan sääntönä pelaajalle on viestien lähettäminen, koska pelaaja pelaa peliä lähettämällä viestejä pelihahmolle. Pelaaja voi itse päättää kirjoittamansa viestin sisällön, joten tältä osin mekaniikkaa ei ole rajoittava tekijä. Mekaniikka on enemmänkin kannustava ja seuraamuksellinen. Koska pelaajan viestien sisältö ei ole rajoitettu, voi pelaaja kokeilla ja tutkia, minkälaisia vastauksia pelihahmot antavat. Pelin vaste pelaajan viesteihin on kumminkin rajattu kehotteen ja pelaajan viestien avulla. Pelihahmon eli kielimallin vastauksia ohjataan kehotteella antamaan pelaajan muutosta vastustavia vastauksia ja pelihahmo ottaa huomioon myös pelaajan lähettämät viestit vastauksissaan.

Argumentointiin perustuvaa pelimekaniikka kuvastaa parhaiten verbi ”keskusteleminen”, koska se on mitä pelaaja tekee pelissä, eli keskustelee pelihahmojen kanssa. Tämä mekaniikka on myös pelin ydinpelimekaniikka, koska pelaaja tarvitsee sitä jatkuvasti pelissä halutun tilan saavuttamiseen, eli tässä tapauksessa saamaan pelihahmot hyväksymään pelaajan muutos.

Arvioin argumentointiin perustuvan pelimekaniikan toimivuutta luvussa 2.4 esitettyjen pelimekaniikkojen päämäärien mukaisesti. Tavoitteen selkeyden osalta pelimekaniikassa on parantamisen varaa. Missään vaiheessa peliä ei kerrota, mikä keskustelujen tavoite on ja selkeää voittotilaa pelissä ei ole. Jokin kehote pelin päämäärästä pelaajalle olisi hyvä selkeyttämään pelin tavoitetta, esimerkiksi viestiruutuun. Voittotila on pelaajan päätettävissä, koska mikään tila pelissä ei muutu, kun pelihahmo hyväksyy muutoksen. Pelaajalle tämän pelihahmon muutoksen hyväksymisen huomaaminen pitäisi olla helppoa, koska pelihahmo ei enää vastustele muutosta, mutta tämä silti jättää tulkinnanvaraa pelaajalle. Pelimekaniikka tarjoaa haastetta pelaajalle, koska pelaajan täytyy osata perustella muutoksen tarpeellisuus pelihahmolle, tai vähintäänkin miksi pelihahmon esittämät argumentit ovat väärässä. Jokaiseen keskusteluun on

mahdollista löytää ratkaisu, joten mekaniikka on tasapainossa. Vuorovaikutus näkyy pelihahmon vastausten muodossa. Pelihahmon vastaus pohjautuu pelaajan antamaan aiheeseen, sekä aikaisemmin lähetettyihin viesteihin, joten pelaajan valinnoilla on myös jokin merkityksellinen seuraus. Pelihahmojen antama palaute on pitkälti negatiivista, mutta se kuuluu tähän pelimekaniikkaan, koska pelihahmon kuuluu vastustella pelaajan muutosta. Pelihahmot antavat kyllä positiivista palautetta, jos ne saadaan hyväksymään pelaajan muutos.

5.2 ChatGPT:n käyttö pelihahmojen vastausten generoimisessa

ChatGPT on hyvä luomaan ihmiselle luettavaa ja ymmärrettävää tekstiä. Paras käyttötarkoitus ChatGPT:lle peleissä olisi generoida pelihahmon vastaukset antamalla pelihahmolle jokin taustatarina, jonka perusteella vastaukset generoidaan. ChatGPT:lle voidaan esimerkiksi kertoa pelihahmon nimi, mitä hän on tehnyt, tekee tällä hetkellä, mitä pelaaja on tehnyt, mitä pelaajalle täytyy kertoa tai minkälaista henkilöä ChatGPT esittää, kuten tein tämän opinnäytetyön toiminnallisessa osuudessa. Näin ChatGPT osaa generoida tilanteeseen sopivia vastauksia pelaajan kanssa keskustelun aikana. ChatGPT:n avulla voidaan myös vähentää pelihahmojen vastauksien toistuvuutta. Pelihahmojen vastaukset eivät olisi täysin ennalta kirjoitettuja ja näin saataisiin vaihtuvuutta pelihahmojen vastauksissa, koska samat vastaukset eivät toistuisi useampaan kertaan.

ChatGPT:tä tai muita vastaavia kielimalleja hyödyntäessä peleihin täytyy myös ottaa huomioon niiden jättämä hiilijalanjälki. Kielimallien kouluttaminen ja käyttäminen vaati energiaa ja riippuen tavasta, jolla energia tuotetaan, vaikuttaa hiilijalanjälkeen.

5.3 Argumentointiin perustuvan pelimekaniikan hyödyntäminen

Pelihahmot usein esittävät jotain toisia vaihtoehtoja muutokselle tai esittävät argumentteja, miksi muutos ei olisi hyväksi. Näitten pelihahmon argumenttien

pohjalta pelaaja voi pohtia, onko muutos tarpeellinen tai onko jotain muuta, parempaa tapaa toteuttaa muutos. Jos pelaaja näkee muutoksen tarpeelliseksi, pelihahmon pystyy kyllä taivuttelemaan pelaajan puolelle, mutta samalla pelaaja saa toisia näkökulmia muutoksesta pelihahmon argumenttien muodossa.

5.4 Tekninen toteutus

Kehotteessa on parantamisen varaa. Tällä hetkellä kehote tuottaa halutunlaisia vastauksia sopivalla syötteellä pelaajalta, mutta ei ota huomioon, jos pelaaja lähettää aiheesta eroavan viestin. On mahdollista, että pelaaja ohjaa viesteillään kielimallin antamaan ei toivottuja vastauksia. Joten kehotetta täytyisi muokata ottamaan huomioon aiheesta eroavat viestit.

LatentActionin käyttäminen HTTP-pyyntöjen aikana oli toimiva ratkaisu, koska HTTP-pyyntöt eivät tapahdu välittömästi, vaan niiden käsittelemiseen kuluva aika voi vaihdella. LatentActionin avulla voidaan odottaa, että viesti on saapunut ja kun viesti on saapunut, voidaan jatkaa muita toimintoja viestin näyttämiseksi.

Unreal Enginen http- ja json -moduulit auttoivat huomattavasti HTTP-pyyntöjen tekemisessä sekä pyyntöjen sisällön luomisessa ja vastauksien käsittelyssä. Moduulien valmiit luokat ja niiden sisältämät funktiot mahdollistivat helpon toteutuksen HTTP-pyyntöjen tekemiseen.

5.5 Vertailu AIDungeoniin

Tässä opinnäytetyössä kehitetty peli ja AIDungeon käyttävät molemmat ChatGPT:tä pelimekaniikan perustana. Yhtäläisyyksiä peleissä on aiheen määrittäminen ennen pelin aloittamista. AIDungeonissa määritellään tarinan aihe ja kehittämässäni pelissä määritellään muutoksen aihe. Suurimpana erona on, että AIDungeonissa luodaan tekstipohjainen tarina pelihahmojen vastausten lisäksi. Pelkät pelihahmojen vastaukset eivät ole keskiössä, vaan lisäksi keskustelun ympärille kerrotaan tarina.

5.6 Jatkokehitys

Nykyisessä muodossa peli ei ole täysin valmis. Kaikki tarvittavat pelimekaniikat peliin on toteutettu, eli pelihahmojen kanssa keskusteleminen ja liikkuminen. Pelin jatkokehitystä ajatellen peliin täytyy vielä lisätä ominaisuuksia pelin pelattavuuden ja ulkonäön kannalta. Kun pelaaja on määrittänyt muutoksen päävalikossa ja siirtynyt pelikenttään, pelaajalla ei ole mitään keinoa palata takaisin päävalikkoon vaihtaakseen muutosta. Peliin täytyy lisätä valikko, jonka kautta hän pääsee takaisin päävalikkoon vaihtaakseen muutosta. Peliin pitäisi lisätä myös ominaisuus, jonka avulla pelaaja pystyisi tyhjentämään keskustelut ja aloittamaan pelin alusta samalla aiheella. Pelikenttä ja pelihahmot ovat myös kehittämisen kohde, kenttä on yksinkertainen, pelihahmot ovat palloja ja pelaajan hahmo on kuutio. Pelin ulkonäön kannalta pitäisi rakentaa uskottavampi pelimaailma ja lisätä pelihahmoille kunnolliset mallit ja animaatiot. Pelillä ei ole myöskään selvää voittotilaa, joten voittotilan lisääminen lisäisi pelattavuutta ja mahdollistaisi selkeämmän tavoitteen pelissä. Tämä vaatisi kehoitteen uudelleen suunnittelemisen, jotta se mahdollistaisi selkeän voittotilan saavuttamisen pelihahmon viestin avulla. Eli peli tarvitsee jonkin mekanismin, joka tunnistaa pelihahmon hyväksyvän muutoksen ja tämä vaikuttaisi pelin tilaan. Tämän voisi toteuttaa avainsanan avulla, eli kun pelihahmo hyväksyy muutoksen ja sanoo tietyn sanan, muutos on hyväksytty ja pelaaja voi siirtyä seuraavaan keskusteluun. Tällä hetkellä minulla ei ole aikeita julkaista peliä, ainakaan nykyisessä tilassa.

Lähteet

- Adams, E. & Dormans, J. 2012. Game Mechanics: Advanced Game Design. Yhdysvallat: New Riders. O'Reilly for Higher Education. 12.10.2023.
- AI Dungeon. 2019. Latitude.
- Bloons TD 6. 2018. Ninja Kiwi.
- Boller, S. 2013. Learning Game Design: Game Mechanics. Knowledge Guru. <http://www.theknowledgeguru.com/learning-game-design-mechanics/>. 25.9.2023.
- Brazie, A. 2022. Video Game Mechanics: A Beginner's Guide (With Examples). Game Design Skills. <https://gamedesignskills.com/game-design/video-game-mechanics/>. 28.9.2023
- Call of Duty 4: Modern Warfare. 2007. Activision.
- Cook, D. 2006. What are game mechanics?. Lost Garden. <https://lostgarden.home.blog/2006/10/24/what-are-game-mechanics/>. 27.9.2023.
- Detroit: Become Human. 2018. Quantic Dream.
- Epic Games. 2023a. Unreal Engine 5. <https://www.unrealengine.com/en-US/unreal-engine-5>. 12.11.2023.
- Epic Games. 2023b. Programming and Scripting. <https://docs.unrealengine.com/5.3/en-US/unreal-engine-programming-and-scripting/>. 12.11.2023.
- Epic Games. 2023c. Gameplay Framework. <https://docs.unrealengine.com/5.3/en-US/gameplay-framework-in-unreal-engine/>. 18.11.2023
- Escape from Tarkov. 2017. Battlestate Games.
- Fabricatore, C. 2007. Gameplay and game mechanics design: a key to quality in videogames. Research Gate. https://www.researchgate.net/publication/236168267_Gameplay_and_game_mechanics_design_a_key_to_quality_in_videogames. 28.9.2023.
- Fullerton, T. 2014. Game Design workshop, 3rd edition. Yhdysvallat: A K Peters/CRC Press. O'Reilly for Higher Education. 12.10.2023.
- Grand Theft Auto V. 2013. Rockstar Games.
- Griesemer, J. 2011. Definition: Game Mechanics. Rewarding Play. <https://rewardingplay.com/2011/01/24/definition-game-mechanics/>. 27.9.2023.
- Halo: Combat Evolved. 2001. Microsoft Game Studios.
- Heikkonen, J. 2018. Interaction in video game dialogue: Exploring the effects of ludic context on fictional dialogue. Helsingin yliopisto. Humanistinen tiedekunta. Pro Gradu -tutkielma. <https://helda.helsinki.fi/server/api/core/bitstreams/6a35ad06-3e64-4238-990f-9c4ecdd2cce6/content>. 24.11.2023.
- JSON. 2023. Introducing JSON. <https://www.json.org/json-en.html>. 18.11.2023
- Järvinen, A. 2008. Games without Frontiers. Tampere University. Väitöskirja. <https://urn.fi/urn:isbn:978-951-44-7252-7>. 6.10.2023.
- Kirkpatrick, K. 2023. The Carbon Footprint of Artificial Intelligence. Communications of the ACM.

- <https://cacm.acm.org/magazines/2023/8/274925-the-carbon-footprint-of-artificial-intelligence/fulltext>. 23.11.2023.
- Klappenbach, M. 2021. What Is a Platform Game?. Lifewire.
<https://www.lifewire.com/what-is-a-platform-game-812371>.
2.10.2023.
- Latitude. 2023. AI Dungeon. <https://aidungeon.com/>. 24.11.2023.
- OpenAI. 2023a. Text Generation. <https://platform.openai.com/docs/guides/text-generation>. 12.11.2023.
- OpenAI. 2023b. Prompt engineering.
<https://platform.openai.com/docs/guides/prompt-engineering>.
12.11.2023.
- OpenAI. 2023c. Pricing. <https://openai.com/pricing>. 12.11.2023.
- OpenAI. 2023d. API reference. <https://platform.openai.com/docs/api-reference>.
15.11.2023.
- Polydin Studio. 2023. Game Mechanics. <https://polydin.com/game-mechanics/>.
18.10.2023.
- Sicart, M. 2008. Defining Game Mechanics. Game Studies.
<https://gamestudies.org/0802/articles/sicart>. 29.9.2023.
- Super Meat Boy. 2010. Team Meat.
- The Binding of Isaac: Rebirth. 2014. Nicalis.
- W3schools. 2023. C++ Structures.
https://www.w3schools.com/cpp/cpp_structs.asp. 16.11.2023.