UNIVERSITY OF MALAYA

The Leader in Research & Innovation

WIA3002

Final Year Project

Blockchain Based VR Multiplayer Mobile Shooter Game

Supervised by:
Senior Lecturer. DR. ALI MOHAMMED MANSOOR ALSAHAG

Prepared by:
Bisher Dokkmak & AbdlRahman Mohammed Fathy Essa

1. Abstract

This project aims to provide a proof of concept to the potential advantages of using Blockchain and VR technologies, two technologies that could be very distinctive outside gaming context. We aim to do so through the development of a simple shooter game that leverages the power and distributivity of Blockchain technology to create scarce non fungible assets, increase player engagement through VR.

The vision is to help visualize a gaming reality where players own their game assets and can make use of them outside of games. In this reality, games have well documented public APIs, and it’s easy to build tools for the gaming community. Even more the assets acquired in one game can be traded/exchanged with assets from or even used directly in gameplay in games other than the one they were first created for. All of this can be brought to the gaming table for free by building games on blockchain.

By using blockchain based personal players’ items, and granting access only to the owner to modify the item in future upgrades or customization securely, the initial model of the game we will be developing will feature VR and Image Recognition elements to engage the player in a tangible experience; and use real-world items and movement to drive the virtual world, creating a more exciting gameplay experience.

The scope of this project is limited to developing the core game mechanics only, using VR and AI SDKs and integrating them into the system, Implementing the Blockchain technology and integrating it with a multiplayer server, which should be bound to one cost-efficient server, as this is only a base project and a proof of concept and might not be well suited to handle a large network of players.

1. Introduction to the relevant project title

Mobile and online gaming are a huge industry that is growing larger day by day with the number of gamers increasing hourly worldwide. With new tools and platforms created regularly; gaming companies and developer teams are constantly trying to come up with more innovative ways to develop games that are exciting, novel and even addictive in order to coup with the increasing demand for such games.

Many of the game projects that are evolving are mass adopting the use of rising technologies as VR and Image Recognition to create enthralling gaming experiences which we believe will set the standards and shape of the future of gaming.

Also with blockchain being a relatively new and promising technology given its potential to transform the industry and bring upon completely new possibilities and chances; we view the understanding and implementation of these technologies to be necessary for us as young students and software developers; which makes us keen to take the initiative to learn and implement these technologies in our project and hence we chose our idea for the game to be simple, just a normal shooting game as a start.

3. Project Objective

The project objective can be wholly summarized in the following points:

- To Develop a Blockchain based VR Multiplayer Mobile Shooter game, Featuring VR, Hand Tracking and Blockchain.
- To integrate an untethered affordable VR module
- To develop a low-cost mobile hand tracking control
- To adopt blockchain technology with game assets
- To evaluate and test the developed system

Thus the final product of this project is aimed to be a functional shooter game with the features and main functionalities brought in by the integration of the previously mentioned technologies and that matches the below descriptions:

(The features that may be discussed here are the most important or dominant parts of the product, that may be implemented as different packages in the architectural diagrams)

3.1.1 VR Capability

Displaying the output using a VR engine to be able to use any VR goggles to play on the player's mobile phone, which increases the player's engagement and experience. The ultimate reason for choosing such technology is the need of showcasing games that implement VR into a real gaming experience, without needing any expensive or additional gadgets.

As this project is developed by two co-developer students, it is specified by University of Malaya's guidelines to have two separate parts or modules of the project, that for each one student should have a substantive role or contribution, which makes this part of significant important, and would be ranked as a High Priority.

However, the risk that comes with the benefits of using such technology needs to be discussed, as VR may increase rendering costs, cause additional overhead for users that may bare from playing because of the additional requirements that may decrease accessibility, in spite of that, this project is developed for academic reasons with some business goals targeting developers, who will not be affected by such concerns. Thus, we would rate the risk as 3 Low.

Stimulus/Response Sequences

The stimulus that may define the interaction with this feature is the display, which is continuous and passive, as it does not change behaviour as a response for a specific user action.

Functional Requirements

REQ-1: The VR API should be easily integratable with Unreal Engine 4.

REQ-2: The VR API should work simultaneously with any supported device. The supported devices would be included in the future <TBD>.

REQ-3: The VR headset should support device's camera, as it would be used in the following features.

3.1.2 Hand and Controller Tracking

Description and Priority

An AI Computer Vision Hand and Object Tracking module that uses a Computer Vision SDK, such as TensorFlow or OpenCV, for the purpose of providing a VR input to the game. The module would use object detection to track the movement of the joystick controller as the gun/holdable items inside the game, which provides more intractability and a richer experience for the players, as well as increasing accessibility.

In addition to what was discussed in the VR Capability feature module priority description, this module also inherits the same high priority, as it is essential for the development of a complete input and output interaction system via VR and CV.

The implementation of this module might pose some processing overhead, which would drastically affect gameplay and make the game unplayable, the thing that would give this module a risk of 6.

Stimulus/Response Sequences

When the user starts a game session, the mobile device's camera starts recognizing the surrounding area for spatial recognition and input to the system, to identify player's motion, such as jumping, crouching, leaning and other motions. The camera supplies an image recognition module feed to recognize the position of the player's hands to track their motion and aim, giving the player full control over the direction and aim of their gun. The module also tracks gun location for position and placing with in the virtual world. Other gestures and functionalities might be included under this module, such as reload, giving-away items or throwing them.

Functional Requirements

REQ-1: The CV API should be integratable with UE4.

REQ-2: This module should not affect performance of the game, methodology should be discussed during development<TBD>.

REQ-3: The module should be able to track hands in real-time, which is of a high priority. However, controller tracking may be inefficient and if so, may be discarded in future phases.

3.1.3 Blockchain Tradable Items

Description and Priority

When we are talking about implementing a blockchain based game, we are talking about relying on this data structure to store, retrieve and add game assets. To relate the previous with our game idea, we want each player to completely own their item through the implementation of wallets, that stores all of the player's items and possibly items from other games too and give the player exclusive to edit or trade those items.

All the guns, armors, explosives, items are blockchain tradable items, unique in the game world, and are tradable as owners of an item can be changed if a purchase/exchange/dispose transaction happens.

Stimulus/Response Sequences

When the player pickups a weapon a new transaction will be added to the blockchain, from the base weapon class without any additions, after each customization the player makes, defeat, win or upgrade during gameplay, which triggers a change request to the block in the blockchain server, which will change the following block and also subject to the consensus algorithm of the used blockchain platform.

System Functional Requirements:

REQ-1: Register new users and initialize a player wallet.

REQ-2: Hold all Unique and non-replicable assets owned by this player inside their wallet using Blockchain.

REQ-3: New players should be able to create a new weapon.

REQ-2: Modify, Transfer & Dispose assets owned by player securely through the use of Blockchain technology.

REQ-3: Assets should have unique physical appearance inside the game.

REQ-4: Ownership is changeable between users.

REQ-5: Protect player exclusive rights to modify/ sell/ dispose their assets through implementation of smart contracts.

3.1.4 Multiplayer Gameplay

Description and Priority

Have an online game server to support multiplayer capabilities, such as teams and online gameplay. As the game is a blockchain-base multiplayer-mode game, having such capabilities is crucial to the development of the game project, thusly, this feature is a High priority.

The risks of having a mal-functional or erroneous game server are not as severe as other more important features, such as blockchain-basing or VR display, thus the risk is ranked as 4.

Stimulus/Response Sequences

When the player joins a game server, he is spawned within the team that he chose when registering to the server, the server keeps the player data in the database of the server. While in game, the server synchronizes player's data between them and keeps them playing in the same world in real-time.

Functional Requirements

REQ-1: Server must synchronize gameplay between players.

REQ-2: Server must only synchronize data necessary for gameplay, to keep the connection optimized.

REQ-3: VR, Blockchain, the server and other features must not affect each others behaviour.

4. Literature review

The BlockChain technology can be utilised in various fields today, even so, like the internet the implementations are so vast that they cannot be thought of today. Just like when the internet, not many would have thought of social networks and online gaming even though the idea of electronic mail then was still a fantasy. It's really early in the blockchain gaming evolution. In fact, the blockchain games that exist today were not built by game developers, as evidenced by the quality of existing games. [1]

Outside of gambling and Ponzi schemes, there are perhaps only two dozen credible projects that actually have the necessary talent and experience to get this industry to the next level. Out of them, four teams raised larger series A rounds targeting anywhere between 100,000 and 1,000,000 gamers for each individual game. Clearly, at the level of blockchain infrastructure today, it's not possible. Several of these teams are actively looking for chains that will help them deliver [2].

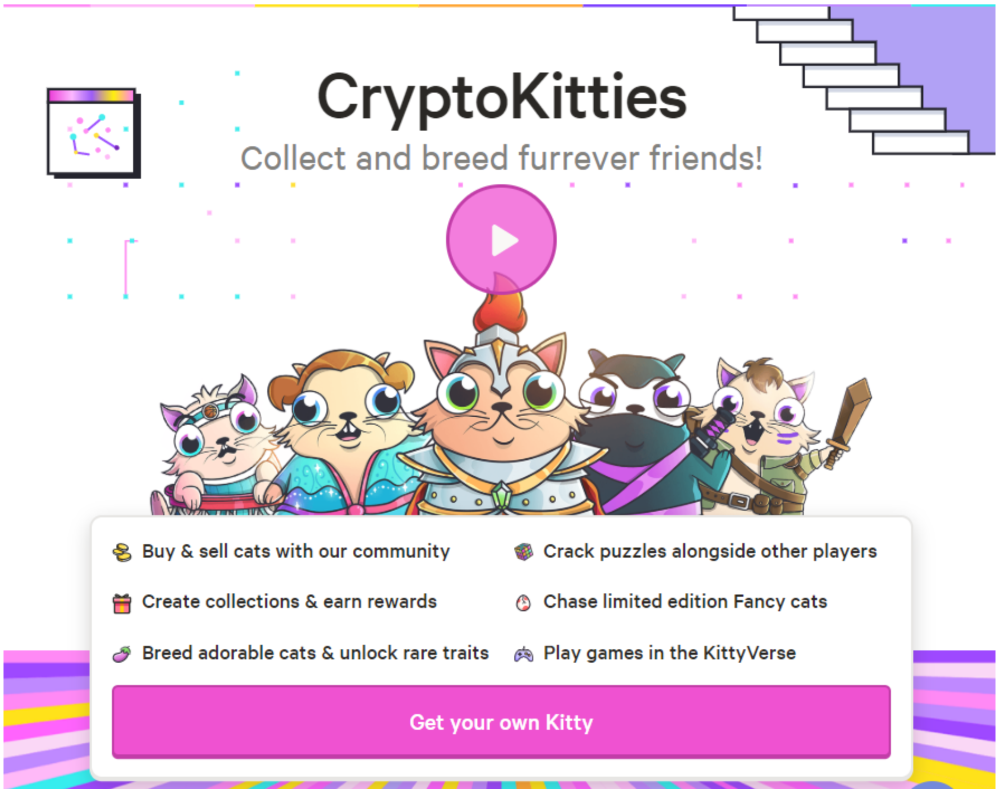
Game developers need to understand how the new way of building games is different from how it’s done today. The blockchain powered backend is slower and more costly but does come with a set of new unique properties, such as social scalability and trust. This new property enables the creation of new markets (where markets didn’t exist prior) and/or replacing networks with markets (which is to say, removing the person in charge of the network and allowing the community to participate in a peer-to-peer exchange of value).

Moreover Scalability is a challenge for teams who aim at getting millions of customers. Gaming executives are excited about the idea of putting a game like WoW with a full economy on blockchain and then pump millions into advertising. That’s the world they want to be in, but that’s not feasible at all today.

What early successful blockchain games witnessed is that there is an emerging behavior of various developers building contracts on top of other contracts, leading to composability and faster innovation. For example, some of the developers who saw the popularity of CryptoKitties decided to build contracts on top of existing DapperLabs contracts, examples being KittyHats and KittyBattles.

Gamers today are used for centralized custody, and it would be good to give them a choice to be able to do centralized custody. Right now, there is no option for centralized custody on existing layer 1 protocols, for instance, Metamask exists for gamers at all times. We are aware of at least 2 really successful games that are moving away from allowing Metamask as a needed step for on-boarding. In any case, optionality for gamers is a must. Compelling gameplay is critical since blockchain games need to balance both new economic model while still providing a utility for the game itself. Even if blockchain brings innovation on economics, the game should still have appeal for a mass gamer. For example, if I go to top apps on DappRadar or State of Dapps, I cannot find a single game with compelling gameplay. [3]

Examples of Existing Blockchain games:

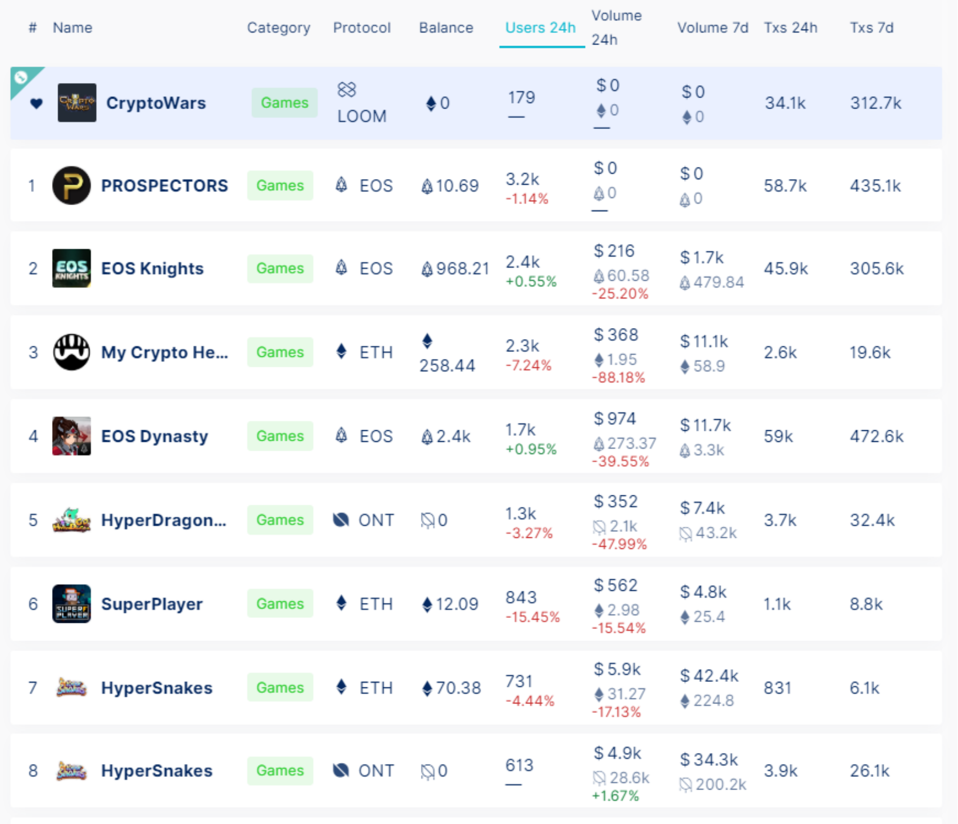


CryptoKitties

Collect and breed furrever friends!

- Buy & sell cats with our community
- Crack puzzles alongside other players
- Create collections & earn rewards
- Chase limited edition Fancy cats
- Breed adorable cats & unlock rare traits
- Play games in the KittyVerse

Most popular Blockchain games on Dapp Radar:



#	Name	Category	Protocol	Balance	Users 24h	Volume 24h	Volume 7d	Txs 24h	Txs 7d
	CryptoWars	Games	LOOM	0	179 —	$ 0 0 —	$ 0 0	34.1k	312.7k
1	PROSPECTORS	Games	EOS	10.69	3.2k -1.14%	$ 0 0 —	$ 0 0	58.7k	435.1k
2	EOS Knights	Games	EOS	968.21	2.4k +0.55%	$ 216 60.58 -25.20%	$ 1.7k 479.84	45.9k	305.6k
3	My Crypto He...	Games	ETH	258.44	2.3k -7.24%	$ 368 1.95 -88.18%	$ 11.1k 58.9	2.6k	19.6k
4	EOS Dynasty	Games	EOS	2.4k	1.7k +0.95%	$ 974 273.37 -39.55%	$ 11.7k 3.3k	59k	472.6k
5	HyperDragon...	Games	ONT	0	1.3k -3.27%	$ 352 2.1k -47.99%	$ 7.4k 43.2k	3.7k	32.4k
6	SuperPlayer	Games	ETH	12.09	843 -15.45%	$ 562 2.98 -15.54%	$ 4.8k 25.4	1.1k	8.8k
7	HyperSnakes	Games	ETH	70.38	731 -4.44%	$ 5.9k 31.27 -17.13%	$ 42.4k 224.8	831	6.1k
8	HyperSnakes	Games	ONT	0	613 —	$ 4.9k 28.6k +1.67%	$ 34.3k 200.2k	3.9k	26.1k

Various Approaches in Blockchain games development; Companies tackling the blockchain gaming space from multiple angles. Some of the teams are focusing on getting non-blockchain indie game studios on board first, while later introducing them to concepts of true digital ownership and ways to implement the marketplace between the games of various game studios. Other teams are focusing on bringing veteran AAA game development teams who built multi-million gamer titles and have them build blockchain games. For such teams, building one game is at least a $1M investment, making them very cautious around their infrastructure choices.

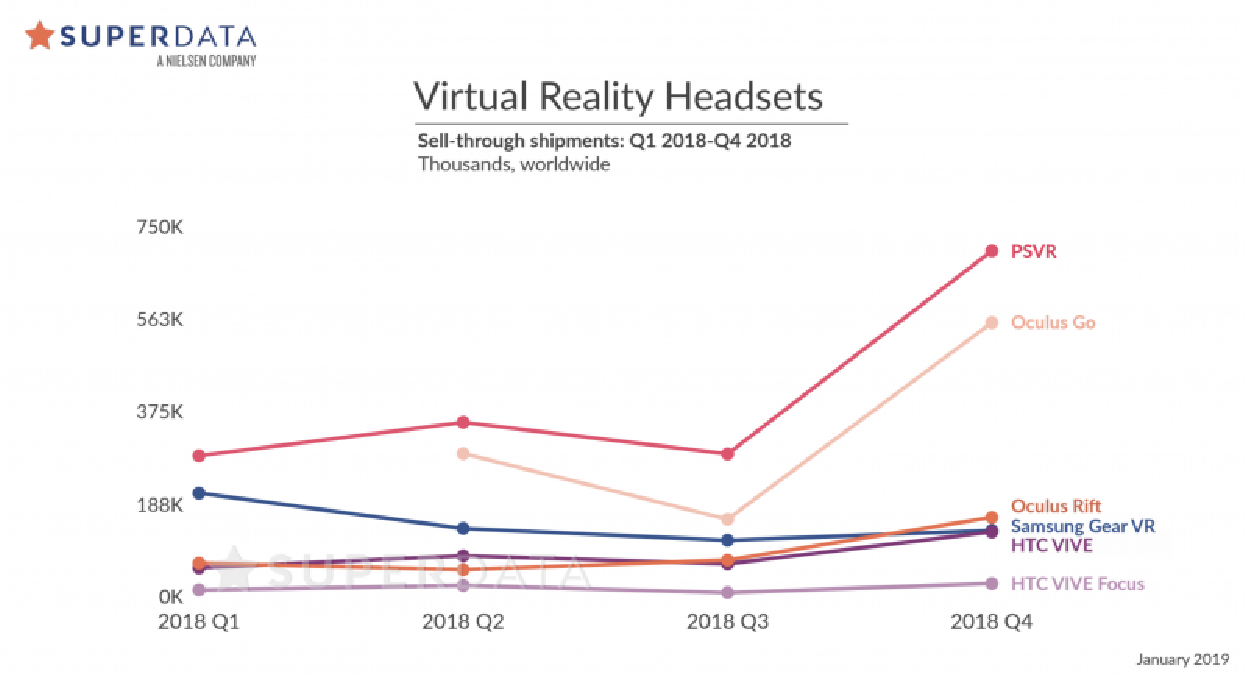
Lastly, some of the approaches involve investing large sums of money trying to identify games with large numbers of engaged gamers and porting them to the blockchain. [4]

When researching VR in games there are multiple platforms that should be discussed and considered, such as HTC Vive, Samsung Gear and Oculus. However, these VR companies and platforms have not been able to achieve market excellence or growth to compete with other gaming markets such as the mobile gaming industry[5], as it had less than 1% revenue share compared to other gaming consoles and platforms. Nonetheless, the VR industry has a regular extreme fluctuation in sales and publicity when a new VR consoles is introduced and commercialized [6]. However, we can identify the reasons for the bad reception of VR platforms and instability in the market and can read a lot of information of the data collected, along with listening and reading the reviews and users' comments about VR games and platforms. On the one hand, VR platforms and accessories are highly expensive and new unique gadgets are not complementary but mandatory for each VR platform and sometimes for specific games. This division in console providers and VR equipments are creating an overhead for users who want to get engaged in the experience only to back off when trying to find the perfect most usable accessory and cost, not with mentioning the substantive weight and room these equipment need, especially since they will not able to be used outside VR with the scarcity and mundanity of games developed for these consoles and equipment, thus making VR a far choice to most common users. Due to the later issue and a number of other considerations such as overhead and complexity of the setup process and the availability of better gaming platforms VR platforms have not had their hit or market demand as it was projected for the past 20 years. Also, a number of medical experts are concerned with the effects VR headsets may have on players playing for too long, while this is not the focus of our project, we should consider this issue when developing our project.

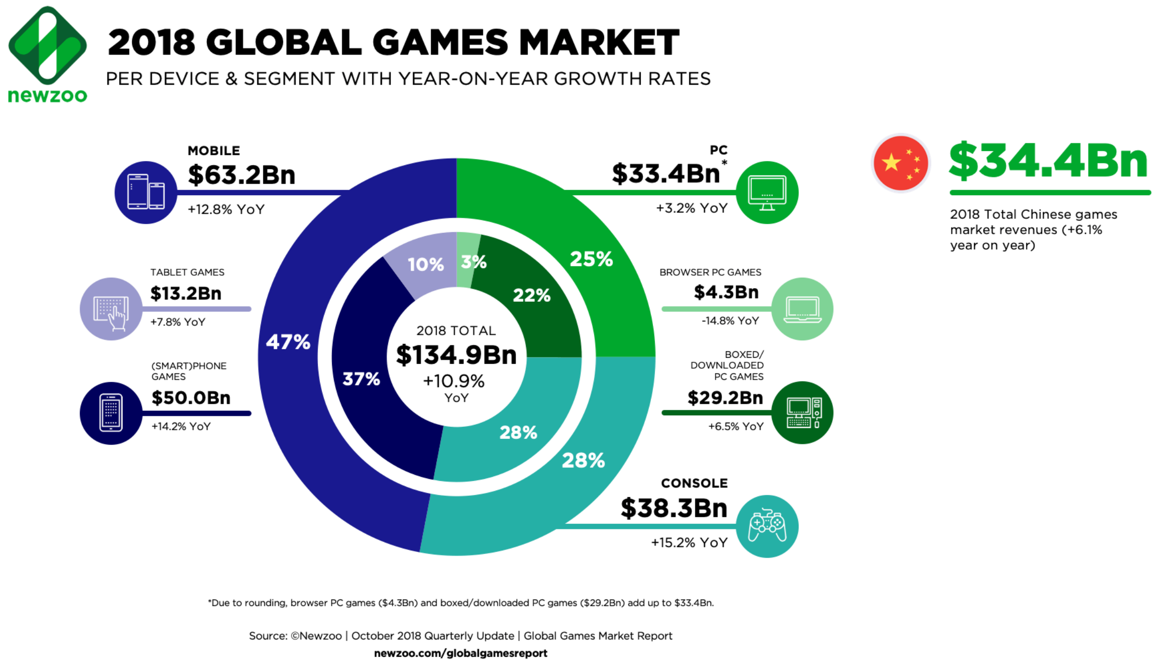
On the other hand, computer platforms and technological advances in AI and algorithms have increased our capabilities in creating software and gave us unlimited resources to what we need and can achieve with our handheld and personal computers [7]. These advances can help us fill the gaps in VR gaming without the need of having to develop further technology when the current technologies and computing power available to every person are not fully used. One of the problems we are having a lag in the VR industry being established in the entertainment and tech industries is the lean to push the boundaries of new technology while ignoring and underappreciating our current technology. However, few companies are trying to bring VR to low and accessible cost and eliminating the need of having to purchase any additional devices or accessories, such as Google developing their VR Cardboard SDK, which allow any mobile device to turn into a VR gear with the need of a simple headset as simple as a cardboard made case, this SDK is also compatible with manyu game engines, giving access for developers to work independently of any brand to easily develop their VR experience with no budget.

Most of the companies are incorporating AI and Computer Vision with in the development of VR consoles or platforms. However, most of the time, purchasing an extra or platform specific gadget is inevitable, an example would be Leap Motion, while the developers are giving a low latency hand tracking module with accurate and efficient gesture recognition, they require the player to purchase their depth sensor which fits a specific number of VR headsets. Manomotion on the other hand, which is an SDK compatible with any mobile device or headset offers a hand tracking module that leverages the power of AI instead of motion sensors to provide a real-time hand control integration for multiple platforms. Other solutions include using OpenCV, which and Open Source Computer Vision library in C++ and Python developed in 1999 and updated every while, which can be used to manually develop a tracking module to be used in games that can remove the use of sensors. However, while OpenCV is a good solution for contour tracking (detection with the use of markers on object), a more suitable and intelligent solution is the use of TensorFlow SDK, which is a Machine Learning library developed by Google for any ML use case, and can be trained for object recognition and tracking.

In the final analysis, the VR industry has a great prospect of growing in the next few years, but cannot achieve that while solely relying on new prospective technological advancements and hardware sensors, and must enhance the current technology and optimize it to leverage its power for a truly accessible and more open VR experience. Achieving relays on the use of many technologies developed in multiple[le areas, integrating these technologies to fix each others faults and help produce a more complete product. After an accessible, easy-to-use, and cross-platform VR solution is produced, game companies and developers would jump into the industry to create a more immersive and engaging experiences and help incorporate VR into the gaming industry. Players would easily be prospected to use VR, as suggested by the fluctuation in the previous graphs, these fluctuations suggest heavy waves of trial and error by the users, which would be resolved and switched into a continuous growth once a solution that would not require any effort or additional cost and gives an experience that competes that of the mobile, PC and game consoles



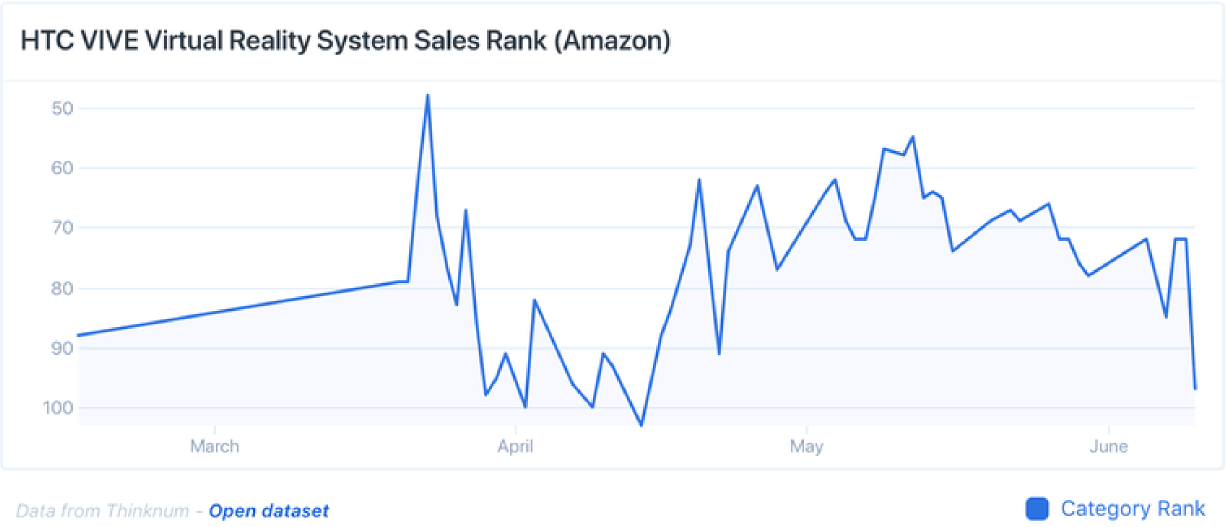
games currently provide is produc



*Due to rounding, browser PC games ($4.3Bn) and boxed/downloaded PC games ($29.2Bn) add up to $33.4Bn.

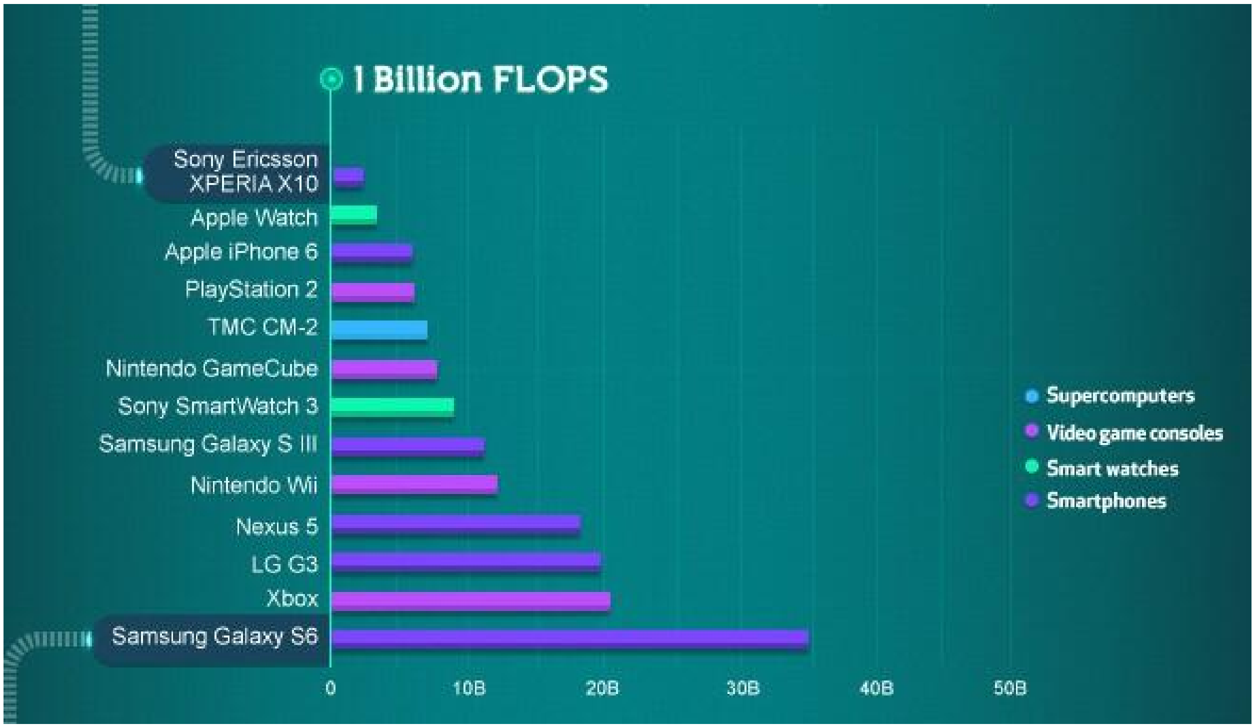
Source: ©Newzoo | October 2018 Quarterly Update | Global Games Market Report
newzoo.com/globalgamesreport

[1] A comparison between VR and Mobile games markets.



Data from Thinknum - Open dataset

[2] HTC Vive sales fluctuation during 2018.



[3] Computer platforms comparison with FLOPS.

5. Problem Statements

- No Reusable game assets in current gaming architectures. The Decentralization brought in by the blockchain based gaming would allow features such as Reusable Game Assets; for example In a normal shooter game like PUBG, the game assets are inaccessible and invaluable outside of the game, however in the case of using blockchain these collectibles live on distributed open database represented by tokens, this means this can be reused outside of the game, and there are marketplaces where you can sell these collectibles; This opens new possibilities as the player is not bound to any particular game and time and value spent is not lost outside the game.

- Absence of unique non fungible assets that belong to each player in most games.

- Most existing Blockchain games lack any real interactive game play. Even if blockchain brings innovation on economics, the game should still have appeal for a mass gamer. For example, if I go to top apps on DappRadar or State of Dapps, I cannot find a single game with compelling gameplay.

- We want to provide an easy to implement, highly usable and reliable solution for games to use VR technology in a more engaging and interactive way, while baring from using extra and more expensive gadgets and accessories.

- The lack of having VR games that leverage the power of VR integrated with hand and object tracking while relying solely on the device processing and functional capability.

- We would propose to solve this by using the device’s camera to detect and map the player’s hands into the virtual 3D world, while also detecting the held joystick as an in-game item.

6. Research Methodology

The blockchain technology is relatively new, that's why finding the right resources and references to aggregate a knowledge base and generate the correct assumptions can be a challenging task given that it's technical complexities and unfamiliarity outside the tech community. Therefore our research rely heavily on findings from pioneering sources on the internet and expert articles.

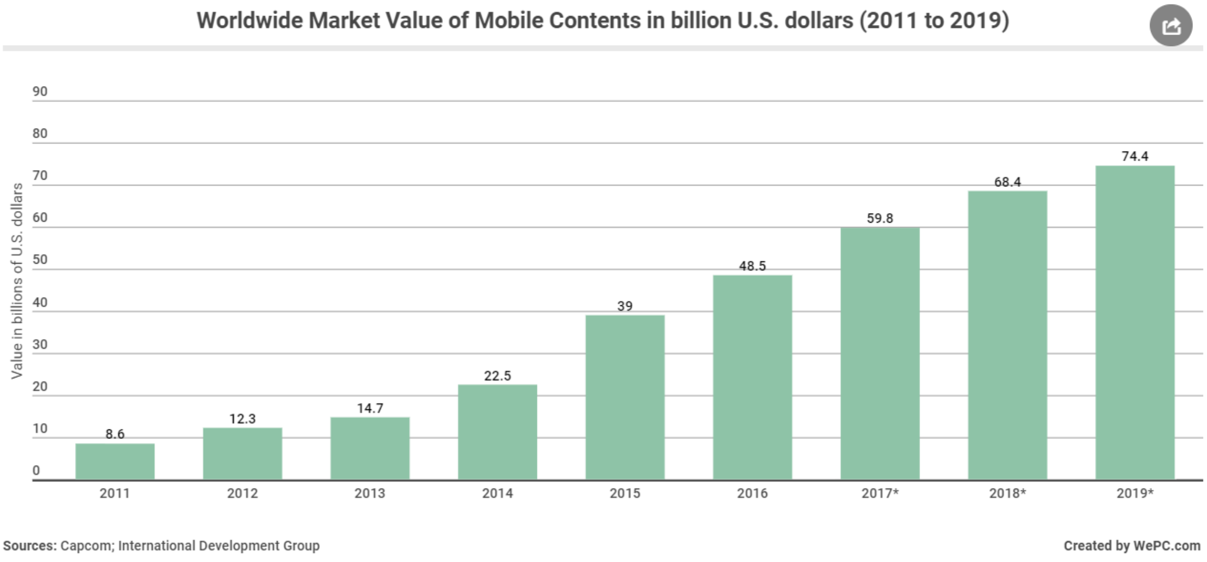
We are keen in our research to use websites that provide statistics and up to date numbers about the gaming industry and the current state of blockchain technologies and trends in blockchain based games.

We reviewed stats and numbers from over many credible websites including: dappradar.com/rankings/category/games, dappradar.com/charts, wepc.com/news/video-game-statistics and many others.

As well as credible content from active professionals and experts from within the gaming industry itself on websites as Medium, hackernoon, github, blockgeeks, etc.

These numbers give clear indications that there is a huge market gap available for the features brought in by the blockchain gaming and there is already a fair demand on these type of games among people who can use the technology.

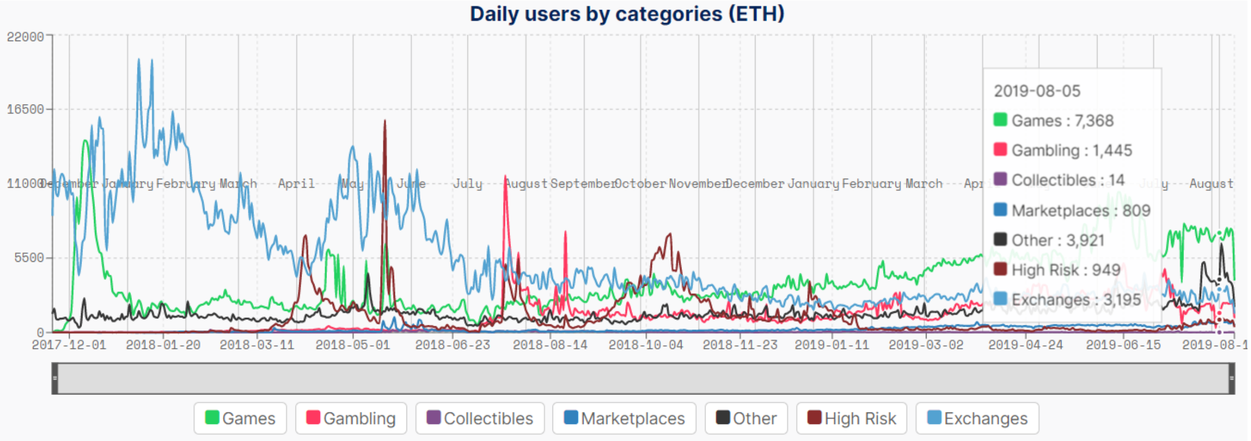
Mobile Gaming



Worldwide Market Value of Mobile Contents in billion U.S. dollars (2011 to 2019)

Sources: Capcom; International Development Group

Created by WePC.com



Daily users by categories (ETH)

In 2017, the market value of the global mobile contents is **59.8 billion U.S. dollars** and is expected to reach **74.4 billion** in 2019. (Capcom, 2017) Tweet Share

United States was the **top country for mobile game session shares**, taking up **20% of the share**, followed by India with 13% of shares. (Techcrunch, 2017) Tweet Share

In 2020, the number of mobile phone gamers in the U.S. is expected to grow by 9% from 2017's record of **192.2 million mobile phone gamers**. (IAB (Trends), 2016) Tweet Share

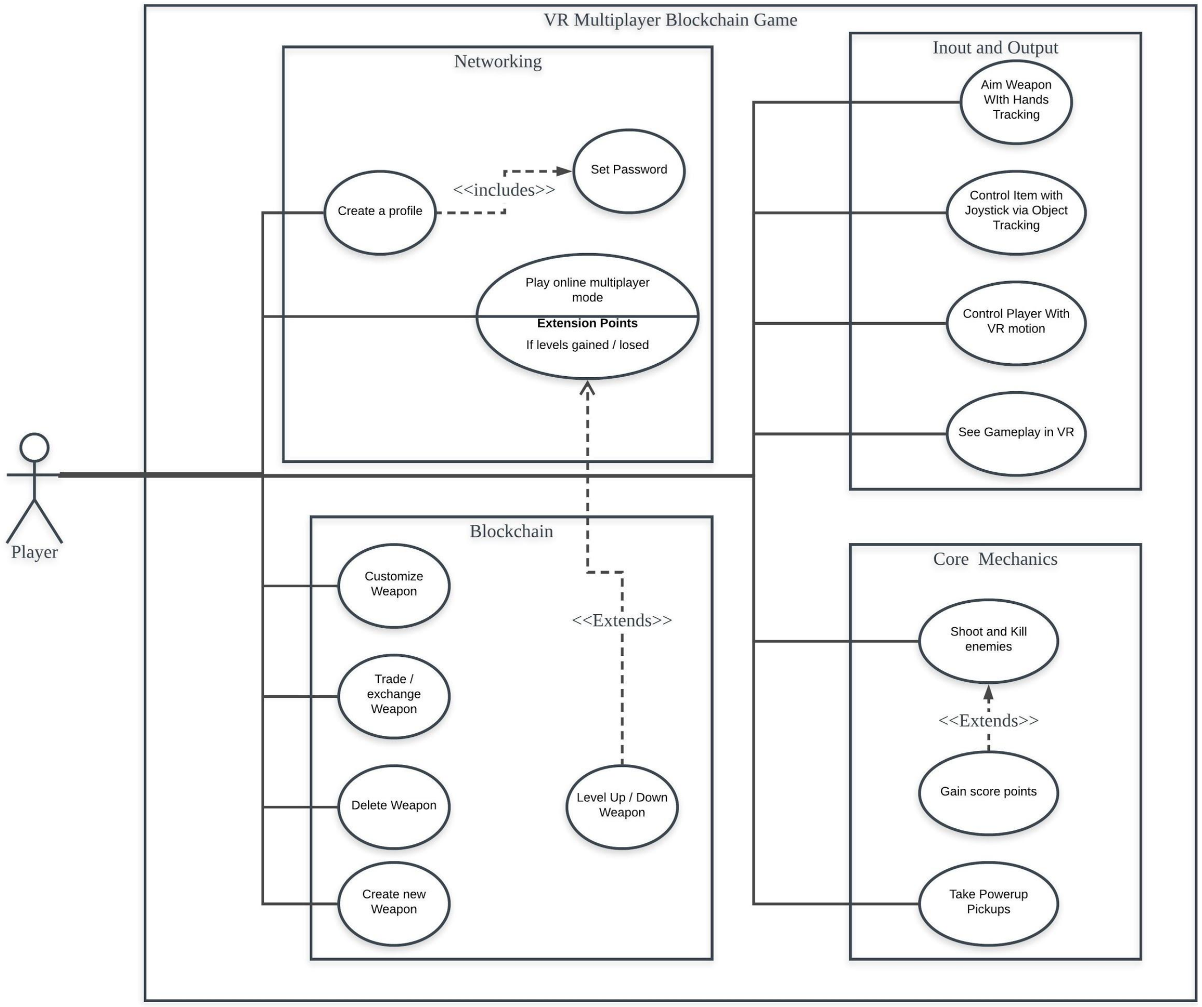
72.3% of mobile users in the U.S. are **mobile phone gamers**. (IAB (Trends), 2016) Tweet Share

7. System analysis and Design

System Architecture:

UseCase Model

The use case diagram showcases the certain features that are implemented in the system inside their respective subsystems. This depicts how the player should and would interact with the system and does not cover the systems interaction with other systems or how it would implement any certain feature.



UseCase	**Description**
UC-1:Play Online	User can use the existent game interface to join a game server
UC-2:Create a profile	User uses the create profile interface to create a new player profile. Having created a profile, user can then join an existing team.
UC-3:Set Password	As a prerequisite of creating and account, user must set a password.
UC-4:Customize Weapon	Users can exercise their right of altering, modifying or exchanging their blockchain based asset.
UC-5:Trade/Exchange Weapon	Users can perform asset transaction of the existent blockchain network, the transaction is recorded in a decentralized electronic ledger and can be viewed by all the other users on this blockchain network.
UC-6:Delete Weapon	Users can delete their item and its record from the entire blockchain.
UC-7:Create New Weapon	Users can create a new base weapon that is yet to be configured, based on their credit score.
UC-8:Level Up/Down Weapon	Items are affected by the player's progress and stats when playing online.
UC-9:Move Hands with Hands Tracking	Using Computer Vision and Image Tracking

	to Recognize player’s hands and move the virtual hands according to the movement of real player’s hands.
UC-10:Control Item with Joystick via Object Tracking	Using Computer Vision and Image Tracking to Recognize the joystick and move the virtual item in accordance to the movement of the joystick.
UC-11:Control Player with VR Motion	Control the movement of the player through the VR sensors, such as crouch, camera rotation and jump.
UC-12:See Gameplay in VR	Use VR Headset on mobile to display gameplay.
UC-13:Shoot and Kill Enemies	One of the core mechanics of the game project is the shooting aspect of the game, being a shooter game, which indicates the presence of Health, Weaponry and Aim.
UC-14:Take Powerup Pickups	Having abilities to pick up during gameplay, such as Health Regeneration and Super Strength.
UC-15: Gain Score Points	When doing certain actions, like killing an enemy, the player’s score should increase based on the credit score of that action.

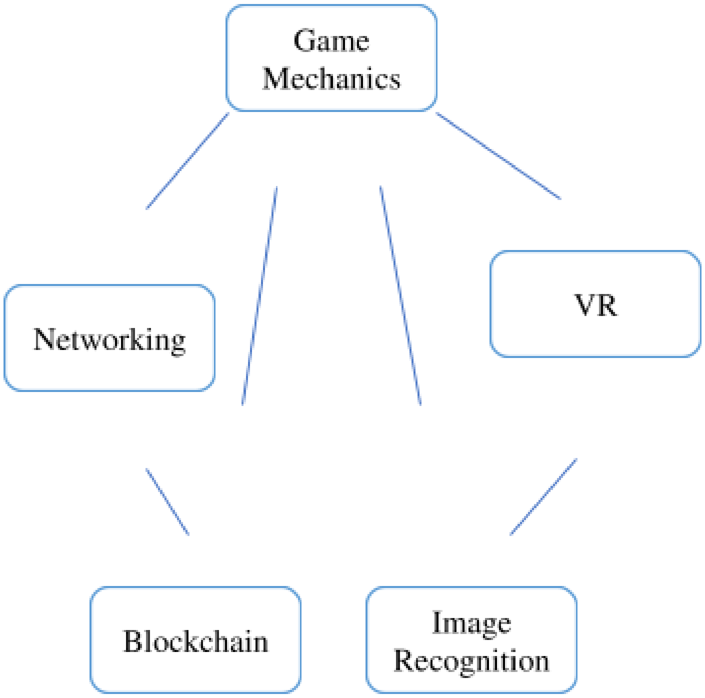
1.1 Product Scope

The software being developed is a game that uses VR for its output, AI Hand tracking and object tracking for its input, Blockchain for distributed asset storage and a server network for multiplayer capabilities. The main goal is to create a proof of concept game of how Blockchain can benefit the game development industry, as Blockchains guarantees uniqueness and distributivity, which has been a long-awaited goal within the gaming industry. Gaming companies are still backing out from this technology and only implementing it for cryptocurrency (see performance requirements section), missing out on a lot of potential use cases that can be acquired from adopting it. Consecutively, our goal is to prove the efficiency of using Blockchain for differentiating players' experiences, which might lead to a space of future applications.

2. Overall Description

2.1 Product Perspective

This product is a standalone first-developed product, developed primarily for this FYP project with the potential to be extended and used as a commercial product after the project is submitted, by the permission of the authors. The project has 5 major parts that need to be integrated for the final implementation, which can be summarized in the following diagram:



2.2 Product Functions

- A **user** registers a new wallet (acconut).
- A **user** must choose a **team** when registering.
- Every **player** is assigned to the **team** he has chosen when registering.
- A **player** can join the **server** to start playing.
- New players can create a new weapon.
- Weapons are stored in player's wallet.
- Every **player** has a set of **upgradeable items**.
- Every **upgradeable item** is unique to the player and initiated in a standard state.
- Every **upgradeable item** is a block in the blockchain.
- **Upgradeable items** characteristics are upgraded/downgraded according to the player's actions.

- **Upgradable items** appearances/looks can be customized by the **player**.
- **Upgradeable items** can be traded or exchanged with other players, while playing or after playing.
- **A Player** can accept or refuse an offer to purchase or trade his **item**.
- **An item** can be discarded and deleted from the blockchain, due to a game mechanic or if the owner **(player)** chooses to.

2.3 User Classes and Characteristics

The system has players as its users, and players are not differentiated by classes, payment subscriptions or plans. However, as in any game project, user classes are referred to as Players' Personas. We can identify multiple main personas of our perspective players, which would allow us to balance the experience between very different player groups.

We identify the following main personas, identified with names for easy referral:

- ***Pro gamer** (Hero_Gun):* *Hero_Gun* here is an expert in multiplayer action games and has a long history of battles, he is a professional gamer in first-person and third-person shooter games, especially when it comes to playing with a team, though he would defeat a team of regular players alone with his eyes closed. *Hero_Gun* might not have high hopes for VR gaming yet, but he is supportive and open to any gaming experience as a pro-gamer, thus he wouldn't be playing the game at all times, and may not be as skilled as he is in other games, but he would consider it as a refreshing and a casual gaming experience.

- ***All things new things*** *(Geekie):* *Geekie* is not as experienced in games as *Hero_Gun*, although he had his share of old games and new games, he was one of the first owners of a Nintendo67 at its time, and keeps every game disk and controller as "old trivia". This guy is down for anything new in the gaming industry and now he is fantasized with the new XR hopes and tries to spend all of his free time on VR and AR games. He is not as proficient as a professional gamer and may not stick to our game for too long, but he would make a portion of our gamers.

- ***Casual Gamer*** *(Paul67):* *Paul67* is a normal person with a job, school or college to attend and work at, his main focus is not on gaming or making a profit out of games. However, he plays games for fun and on the weekends. *Paul67* has purchased a new VR set lately for his mobile would like to try new VR games as his new gaming dose, although he plays only for a bunch of hours a week, and may play a few games, he would give our game a whirl or two every week or so.

- ***Bread to XR*** *(Lava_06):* *Lava_06* is one of the kids that were brought up playing VR and new tech games, he is used to VR games and new experiences games more than old school games, so he would make the most professional player between all of our gamers. Our game wouldn’t be the first of its kind to *Lava_06*, only he and *Geekie* would be the most to appreciate having unique items for each player (use of blockchain). He would play our game as long as it’s supported nearly every day and thus make the most of our population.

2.4 Operating Environment

The game should work on mobiles that would be able to support VR headsets, and that can handle the graphics and processing of the game. The mobile supporting specifications could not be decided upon before testing, but the expected minimum requirements would be idevice 6+, iPad 2017+, and for android 4GB of RAM and 16GB of memory, with the camera resolution of 8MP at least for hand and object tracking.

For the software, any mobile OS that supports VR is acceptable, we expect them to be the same as the above.

SDKs that need to be working on the mobile device are not yet finally specified. However, Vuforia engine is a suspected feature:

iOS Devices:

- *iOS Version: 9+*

Android Devices

- *Android Version: 6.0 (Marshmallow) or newer*
- *IMU w/ Gyroscope Sensors*
- *Selective Image Stabilization*

UWP Devices

- *32 & 64-bit Universal Windows Platform Anniversary Update, or Later*
- *IMU w/ Gyroscope Sensors*
- *Model Targets Minimum Requirements*

iOS Devices

- *Dual-Core CPU*
- *1-2GB RAM (dependent on model size)*
- *iOS Version: 9+*

Android Devices

- *Quad-Core CPU*
- *Memory: 1-2GB RAM (dependent on model size)*
- *Android Version: 6.0 (Marshmallow) or newer*
- *IMU w/ Gyroscope Sensors*
- *Selective Image Stabilization*

UWP Devices

- *32 & 64-bit Universal Windows Platform Anniversary Update, or Later*
- *IMU w/ Gyroscope Sensor*

2.5 Design and Implementation Constraints

As this project is for academic purposes, and as we are trying to achieve a certain point of development that would be sufficient for proving the validity and value of our mission statement, our project may be bound to certain constraints and limitations, to mention a few:

- Having a document SDLC with specific requirements that satisfy the university's FYP requirements.
- Coming up with a result and a document to summarize our mission, how it was achieved, and whether it is fully achieved or not.
- The project should be developed with software engineering standards, documentation, and structure, whereas conventional game development projects use different methodologies and structures.
- Only demo / free or low-cost servers could be used for multiplayer.
- Optimizing the game for mobile devices.
- Optimizing and solving the issues of using blockchain while rendering.
- Having to develop multiplayer ready with blockchain, while both technologies are fairly new to us.
- Collaborating blockchain data structure with other conventional databases and integrating blockchain with the game and server.
- Having a very tight schedule for development and documentation.
- The project and the developed product must have at least two separate modules or sub-products that each of the two developers would develop individually.

2.6 User Documentation

The demanded documentation for this project would be the FYP documentation regulated by Faculty of Computer Science and Information Technology, University of Malaya. Certain pre-specified components are the document that needs to be compiled with the program. However, no user documentation /manuals /tutorials are mandatory for this project.

2.7 Assumptions and Dependencies

These points might be updated in newer versions of the document and changed throughout the development life-cycle:

- Unreal Engine 4.22 is our main game engine, with its embedded framework that uses C++.
- Our main intended platform is mobile gaming.
- The use of Image Recognition with OpenCV or TensorFlow.
- The use of an API that helps with hand tracking, such as Leap Motion.
- The use of an API that helps with the implementation of the Blockchain data structure.
- The use of any server to implement multiplayer, such as AWS, Google Cloud or Microsoft Azure.

3. External Interface Requirements

3.1 User Interfaces

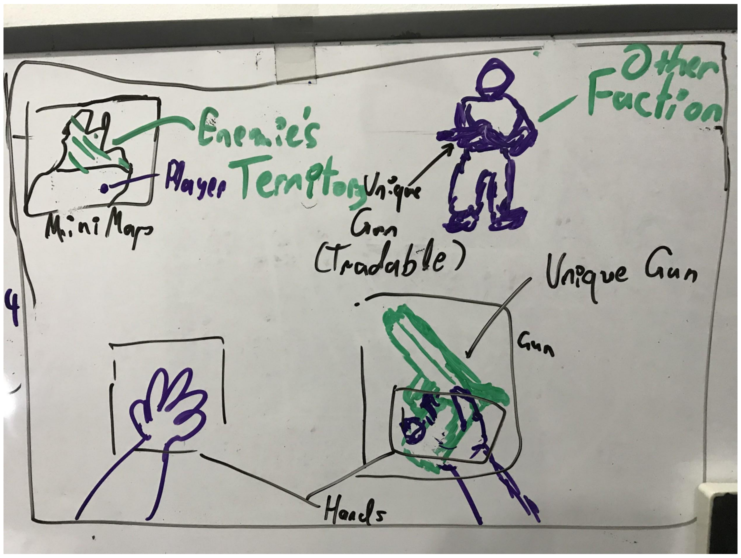
We will discuss the basic layout and components of each main screen the program will have; drawings and illustrations might be included.

Main Menu:

- Career: the main sub-menu where players can join server to start playing or start a new campaign, which resets the player's progress.
- Inventory: provide multiple options for customization and personalization, where players can personalize their items, configure multiple weapon tiers, or customize their character.
- Team: under this sub-menu, players can check their team's stats, controlled territories, members, kills and other numerical characteristics.

- Friends: A way of collaborating with other gamers, may be using another software's API, such as steam or Origin.
- Store: The main place for people to put their developed items for sale or exchange, it may be a place for the developers to put rare or legendary items for sale. May use an in game currency.
- Settings: The main sub-menu for configuring game settings, and should have the following main options:
  - Graphics settings: main configuration for the display and graphics.
  - Sounds settings: configurations for the sound levels and in-game music.
  - Peripheral devices: settings for connecting the VR set and controllers.
  - Control settings: setup key-mapping and player's control preferences.
  - Account settings: this menu is to show the stats of the player's account in our network and to show legal approval documents and privacy regulations, it also gives the player the choice to delete his account or change his display name, email or other details about his account.

Heads-Up Display:

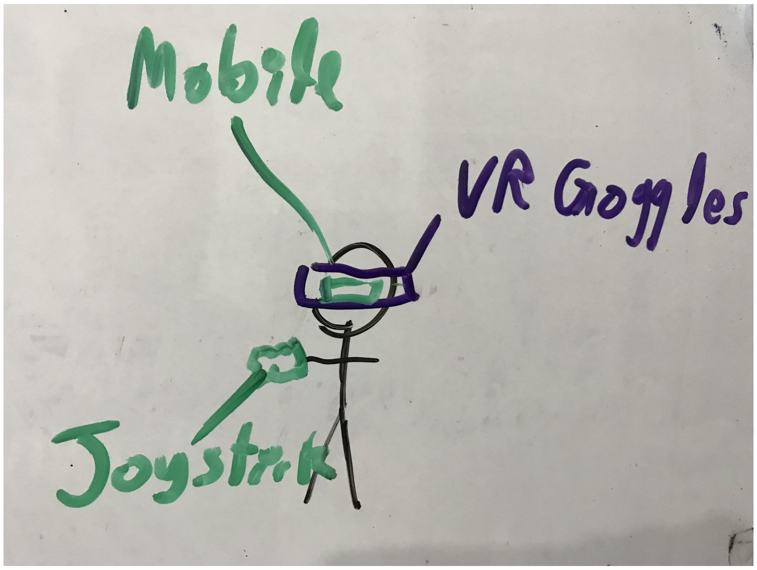


- Mini-map: a minimap to show the player's position and other players, as well as other locations and items.
- Enemies vs Team-mates outline effect: a simple fresnel effect to show a helpers outline that differentiate between teammates and enemies. It also can be used to help players find collectables and items.
- Items' 3D labels: a 3D label that shows informative text above collectables and pickup items, such as available ammo, type of pickup or other useful information.

In-game pause menu:

- Map: opens the full map for the player to set targets or show map information.
- Change weapon tier: Players can set weapon tiers from the main menu, where they can select their pre-built tiers to equip after they die.
- Settings: a small version of the full settings menu from the main menu, contents should be specified during development.
- Trade items: a menu to offer tradable-items for sale or exchange, offers are shown to all players in the server in real-time. Offers here are collaborated with the store.
- Online Players: a menu to show all available online players on the server. Players can offer to trade/exchange their items, or send a friend request.
- Exit: exit to the main menu.

3.2 Hardware Interfaces



- Physical interfaces:
    - Mobile device that supports VR and has a capacity to support the graphics and the use of blockchain that might require some of the GPU for mining in some instances, the mobile device should have storage for game files and blockchain storage. The minimum and preferred requirements for the supported devices will be documented in the final version of the document or the project documentation.
    - Basic VR headset, no current specific requirements are demanded of the VR headset, only that it would fit the mobile device. However, future characteristic may include special cameras for depth and 3D sensing.
    - a Joystick controller to control and interact with the mobile device, this may also be tracked using image tracking to simulate holdable items. However, the controller may be replaceable in future iterations of the project.
- Logical interfaces:
    - Bluetooth controller communication: for the communication between the joystick and mobile device.
    - Camera for AI tracking: for the purpose of Object Tracking the mobile device's camera should be used. This camera may need to support depth sensing.

    - WIFI or 4G/5G network for online connectivity: the mobile device must have an active online connection via WIFI 4G or any other connection.

3.3 Software Interfaces

- Cloud server: any cloud server that gives the functionality needed for this product, such as database storage, multiple players support, integratable API library.
- Blockchain server: Provided by Ethereum rinkeby testnet through Infura API.
- VR display library: for the purpose of supporting VR display. This may be supplied by the UE4's API.
- AI Image Tracking module: a Computer Vision library that supports our development environment. This needs to have a trained model for Hand Tracking and object recognition for joystick detection.
- Networking API: to support players' interactivity in a multiplayer environment. This may be supplied by UE4's API.

3.4 Communications Interfaces

Communication is a defining factor in this project as multiplayer support is a core mechanics, despite the fact that the purpose of this project is the demonstration of a game developed with blockchain as its core mechanic, which heavily depends on networking and communication with a blockchain server.

Below is a list of the main required networking interfaces:

- Communication with the game server: the game server is a server that implements connectivity between players, supports in-game store and stores all game data; which includes players' stats, teams' stats, players accounts, and all game's multiplayer data excluding those related to blockchain. This communication needs to be fast, reliable and secure, only that security is not a major issue for this particular connection.

- Communication with the blockchain server: Integration server, written in Java, which interacts with Ethereum blockchain. Unreal Engine (UE4) Plugin, which sends requests to the Java integration server and getting responses from it. Currently, integration server is using an ethereum node endpoint called Infura to interact with Ethereum blockchain. UE4 Plugin interacts with the integration server via JSON requests via HTTPS.
- Communication with the joystick controller: Communication between the mobile device and the controller should be achieved using bluetooth.

4. System Features

The features that may be discussed here are the most important or dominant parts of the product, that may be implemented as different packages in the architectural diagrams:

4.1 VR Capability

4.1.1 Description and Priority

Displaying the output using a VR engine to be able to use any VR goggles to play on the player's mobile phone, which increases the player's engagement and experience. The ultimate reason for choosing such technology is the need of showcasing games that implement VR into a real gaming experience, without needing any expensive or additional gadgets.

As this project is developed by two co-developer students, it is specified by University of Malaya's guidelines to have two separate parts or modules of the project, that for each one student should have a substantive role or contribution, which makes this part of significant important, and would be ranked as a High Priority.

However, the risk that comes with the benefits of using such technology needs to be discussed, as VR may increase rendering costs, cause additional overhead for users that may bare from playing because of the additional requirements that may decrease accessibility, in spite of that, this project is developed for academic reasons with some business goals targeting developers, who will not be affected by such concerns. Thus, we would rate the risk as 3 Low.

4.1.2 Stimulus/Response Sequences

The stimulus that may define the interaction with this feature is the display, which is continuous and passive, as it does not change behaviour as a response for a specific user action.

4.1.3 Functional Requirements

REQ-1: The VR API should be easily integratable with Unreal Engine 4.

REQ-2: The VR API should work simultaneously with any supported device. The supported devices would be included in the future <TBD>.

REQ-3: The VR headset should support device's camera, as it would be used in the following features.

4.2 Hand and Controller Tracking

4.2.1 Description and Priority

An AI Computer Vision Hand and Object Tracking module that uses a Computer Vision SDK, such as TensorFlow or OpenCV, for the purpose of providing a VR input to the game. The module would use object detection to track the movement of the joystick controller as the gun/holdable items inside the game, which provides more intractability and a richer experience for the players, as well as increasing accessibility.

In addition to what was discussed in the VR Capability feature module priority description, this module also inherits the same high priority, as it is essential for the development of a complete input and output interaction system via VR and CV.

The implementation of this module might pose some processing overhead, which would drastically affect gameplay and make the game unplayable, the thing that would give this module a risk of 6.

4.2.2 Stimulus/Response Sequences

When the user starts a game session, the mobile device's camera starts recognizing the surrounding area for spatial recognition and input to the system, to identify player's motion, such as jumping, crouching, leaning and other motions. The camera supplies an image recognition module feed to recognize the position of the player's hands to track their motion and aim, giving the player full control over the direction and aiming of their gun. The module also tracks gun location for position and placing with in the virtual world. Other gestures and functionalities might be included under this module, such as reload, giving-away items or throwing them.

4.2.3 Functional Requirements

REQ-1: The CV API should be integratable with UE4.

REQ-2: This module should not affect performance of the game, methodology should be discussed during development<TBD>.

REQ-3: The module should be able to track hands in real-time, which is of a high priority. However, controller tracking may be inefficient and if so, may be discarded in future phases.

4.3 Blockchain Tradable Items

4.3.1 Description and Priority

When we are talking about implementing a blockchain based game, we are talking about relying on this data structure for storing our in game assets. To relate the previous with our game idea, we want to create the main weapons and items of our game using ERC-721 tokens, that are owned by the player, the scarcity of these tokens is ensured by the immutable smart contracts.

All the guns, armors, explosives, items are blockchain tradable items, thus are unique in the game world, and are tradable as owners of an item can be changed if a purchase/ exchange/ dispose transaction happens.

4.3.2 Stimulus/Response Sequences

When a player creates a new weapon, this transaction is recorded on the blockchain and ownership of this weapon is transferred to the player's address. After each customization the player makes, defeat, win or upgrade during gameplay, which triggers a change request to the block in the blockchain server, resulting in a new transaction that is also recorded on the blockchain and displayable during gameplay.

4.3.3 Functional Requirements

- Register new users and initialize a player wallet.
- Hold all Unique and non-replicable assets owned by this player inside their wallet using Blockchain.
- Give assets physical presence inside the game.
- Modify, Transfer & Dispose assets owned by player through the use of Blockchain technology.
- Protect player exclusive rights to modify/ sell/ dispose their assets through implementation of smart contracts.

4.4 Multiplayer Gameplay

4.4.1 Description and Priority

Have an online game server to support multiplayer capabilities, such as teams and online gameplay. As the game is a blockchain-base multiplayer-mode game, having such capabilities is crucial to the development of the game project, thusly, this feature is a High priority.

The risks of having a mal-functional or erroneous game server are not as severe as other more important features, such as blockchain-basing or VR display, thus the risk is ranked as 4.

4.4.2 Stimulus/Response Sequences

When the player joins a game server, he is spawned within the team that he chose when registering to the server, the server keeps the player data in the database of the server. While in game, the server synchronizes player's data between them and keeps them playing in the same world in real-time.

4.4.3 Functional Requirements

REQ-1: Server must synchronize gameplay between players.

REQ-2: Server must only synchronize data necessary for gameplay, to keep the connection optimized.

REQ-3: VR, Blockchain, the server and other features must not affect each others behaviour.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

Based on the capabilities of the current phones and the android/IOS systems, performance should not be an issue. However, phones with weaker hardware may incur some difficulties rendering the VR environments and may run slower or crash. The game design will be designed to give an enjoyable experience. The functionality of the game will be simplified enough but not trivial, and the graphics will not be overly detailed as to avoid any hardware crashed or potential slow downs.

5.2 Safety Requirements

5.2.1 Avoiding Simulator sickness

Description: This is similar to motion sickness and may cause nausea, sweating or vomiting. The product will help mitigate this by using correct camera calibration and distortion. Also, a warning label will be printed on the product to address this requirement.

Source: Virtual Reality Device.

Constraints: Different users react differently to VR environments so each user must assess their susceptibility.

Standards: None.

Priority: Critical

5.2.2 Considering Mental Health

Description: A label that warns against excessive use will be affixed to the Virtual Slugger as excessive use could affect the health of the user.

Source: Virtual Reality Device.

Constraints: The system cannot enforce proper use.

Standards: None.

Priority: Moderate.

5.3 Security Requirements

- The system should be able to verify and record each and every transaction made on the peer to peer network.

- The content of the blocks should assure integrity preservation, and every block should contain a message digest of the previous block, the timestamp of its creation and other transactional data.
- The design principles behind Blockchain technology offer resistance to modification of data. It records the secured points between multiple players efficiently in a verifiable and permanent way. Each block is set in chronological order by linking them with the previous block and this makes it difficult to add forged blocks
- Each time a transaction is made, it has to be added to the Blockchain of those players who are involved in the transaction, Thus, a history of all the transactions is maintained using a Blockchain.

5.4 Software Quality Attributes

The game initial build should be able to satisfy the following quality attributes:

- Reliability and correctness by responding to player's input in real time.
- When the player uses hand tracking the game graphics should be able to respond and simulate in real time manner.
- The player should be able to see the results of their input in milliseconds and not after 2 seconds from the effect.
- For adaptability and flexibility, the game will automatically save the player's stats and progress after every level and gaming session.
- In terms of usability the graphical user interface should be intuitive to use, the hand tracking mechanism should be simple and efficient to variety of users.
- Our game shall focus in between ease of use and ease of learning.
- The game should be designed as to be picked up by any player and learned/played easily.
- the game play will ease players into learning and exploring existent commands/features by gradually introducing them and giving tips during the gameplay.
- by the end of a gaming session user should be familiar and able to use most of the features and commands introduced.

System Architecture Summarization:

ID	Quality Attribute	Scenario	Associated Use Case
QA-1	Real-time behaviour	As it’s a multiplayer game, real-timeliness is a core attribute to be satisfied, as the player state should always be updated and synced to other players’ states.	1 - 7 - 9 - 10 - 11 - 12 - 13 - 14 - 16
QA-2	Resource Utilization	Optimization is an important priority when it comes to game development, as this project needs to concurrently provide multiple features and support multiple systems while sustaining a high real-time FPS rate.	1 - 9 - 10 - 11 - 12 - 13 - 14
QA-3	Interoperability	The game should interface with multiple systems, including blockchain servers, social networking and other SDKs.	5 - 6 - 7 - 10 - 11 - 12 - 13
QA-4	Coexistence	The game core systems should coexist with each other and with other systems in order to support robust gameplay.For example, The blockchain mining system should not affect the render engine.	3 - 7 - 9 - 10 - 11 -13
QA-5	Learnability	The learnability line should not be over complicated, as players should easily adapt and use the game controls and understand its features. However, this	1- 4 - 6 - 10 - 11 - 12 - 14 -15 - 16

		should not be limiting to the players' expertises.	
QA-6	Availability	The servers' downtime is crucial to any game that operates online. This should be given attention in terms of future scalability.	1 - 5 - 16
QA-7	Integrity	Data stored on a blockchain block should be kept safe from players' tempering with its data if they don't own the block, access is limited from excessive editing if the player owns the block.	6 - 7 - 8 - 9 - 17
QA-8	Nonrepudiation	As the blockchain network needs to be synchronised and kept non-redundant throughout the network between users, this quality attribute must be met to the highest quality. However, this should be controlled with in specific.	6 - 7 - 8 - 9
QA-9	Authenticity	Blockchain data is kept synced and validated for each user to ensure the safety from any manipulation and protect the data from being tempered.	6 - 7 - 8 - 9

Constraints

ID	**Constraints**
CON-1	The game should run on a minimum 30 Frames per second on regular mobile

	devices (iOS and Android).
CON-2	Player should always be online to play.
CON-3	No additional external devices should be required.
CON-4	All players should be supported on the same server
CON-5	Having the 2 contributors each develop his own subsystem

Architectural Concerns

ID	**Concern**
CRN-1	Establishing a robust control system and VR support
CRN-2	Having the blockchain securely developed and tightly integrated in the system
CRN-3	Establishing a networking session with a functioning blockchain layer

2. Review Inputs

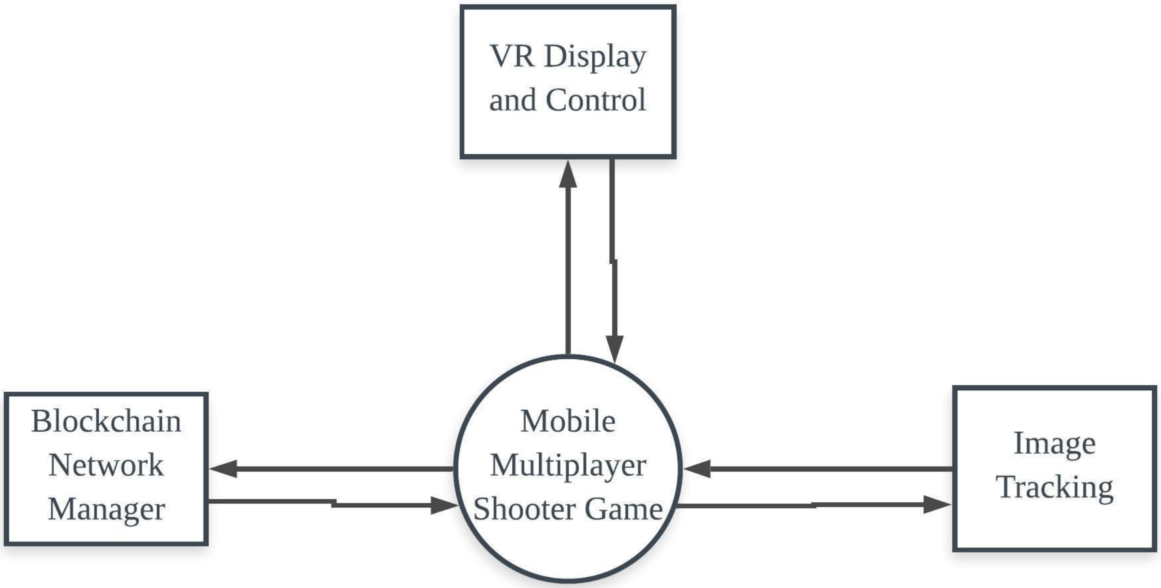
We have summarized the input drivers in the following table:

Category	**Details**
Design Purpose	Establishing a system structure to support the different components and subsystems.
Primary Functional Requirements	The primary use cases are determined to be: UC-7, UC-9, UC-10, UC-11, UC-12, UC-13, UC-14, UC-17
Quality Attribute	

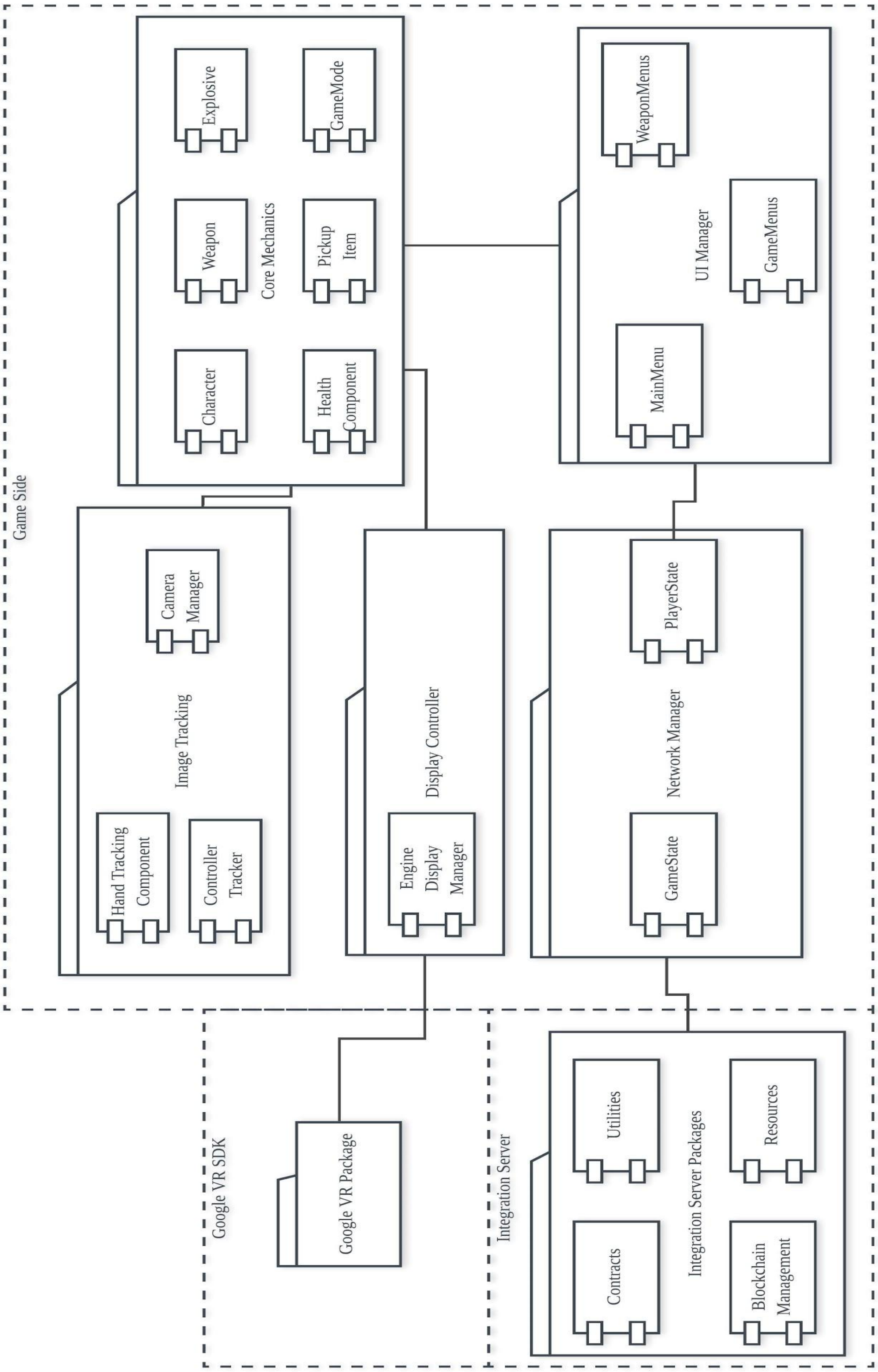
<table>
<tr><td>Scenarios</td><td><table>
<tr><th>ID</th><th>Importance</th><th>Difficulty</th></tr>
<tr><td>QA-1</td><td>High</td><td>Low</td></tr>
<tr><td>QA-2</td><td>Medium</td><td>High</td></tr>
<tr><td>QA-3</td><td>High</td><td>Medium</td></tr>
<tr><td>QA-4</td><td>Medium</td><td>Medium</td></tr>
<tr><td>QA-5</td><td>Low</td><td>Medium</td></tr>
<tr><td>QA-6</td><td>Medium</td><td>Low</td></tr>
<tr><td>QA-7</td><td>High</td><td>High</td></tr>
<tr><td>QA-8</td><td>High</td><td>High</td></tr>
<tr><td>QA-9</td><td>High</td><td>High</td></tr>
</table></td></tr>
<tr><td>Constraints</td><td>As mentioned above</td></tr>
<tr><td>Architectural Concerns</td><td>As mentioned above</td></tr>
</table>

System Design

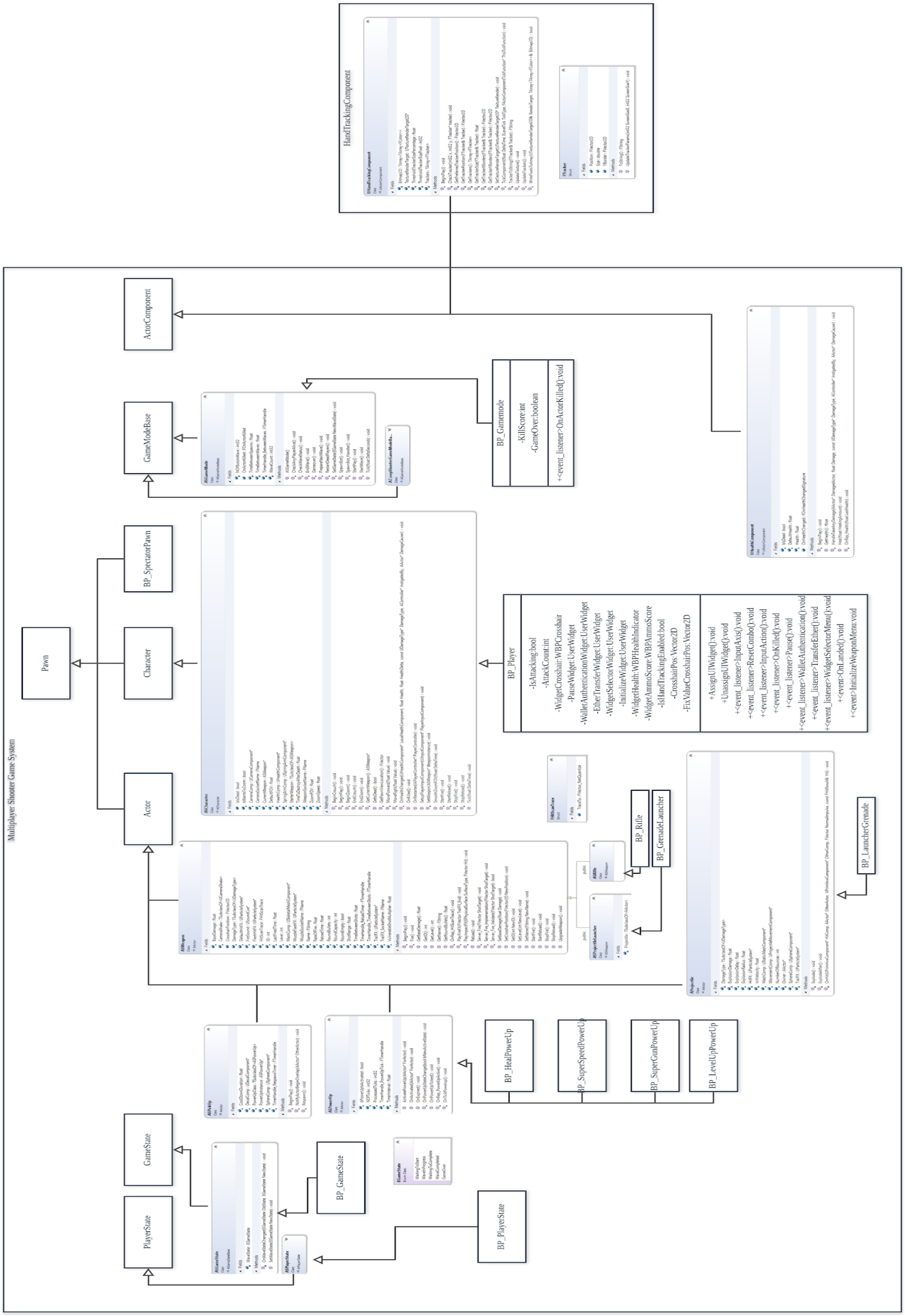
Context Diagram



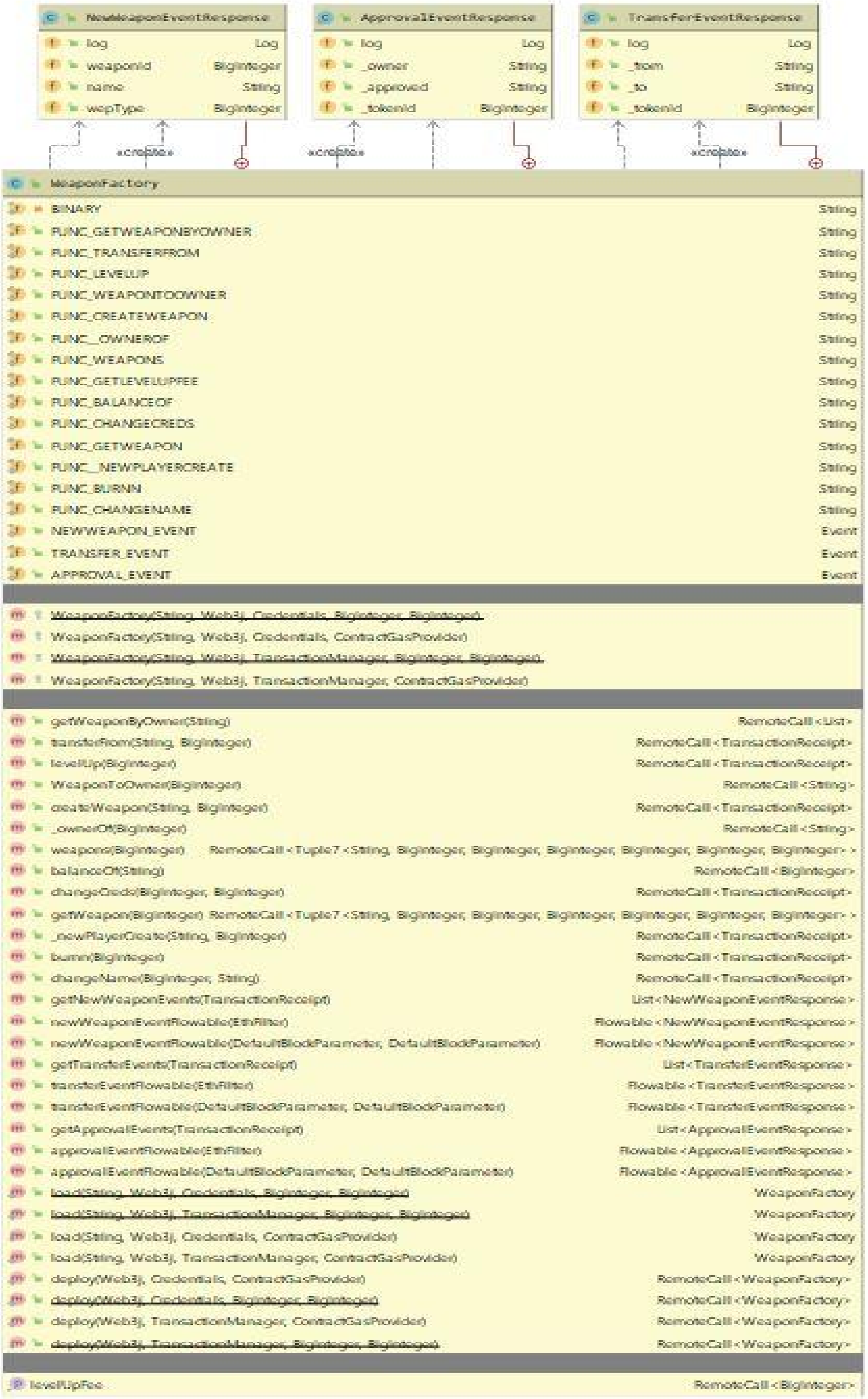
Package Diagram



Class Diagrams


HandTrackingComponent
Multiplayer Shooter Game System
ActorComponent
GameModeBase
BP_Gamemode
-KillScore:int
-GameOver:boolean
+<event_listener>OnActorKilled():void
Pawn
BP_SpectatorPawn
Character
Actor
BP_Player
-IsAttacking:bool
-AttackCount:int
-WidgetCrosshair:WBPCrosshair
-PauseWidget:UserWidget
-WalletAuthenticationWidget:UserWidget
-EtherTransferWidget:UserWidget
-WidgetSelectorWidget:UserWidget
-InitializeWidget:UserWidget
-WidgetHealth:WBPHealthIndicator
-WidgetAmmoScore:WBPAmmoScore
-IsHandTrackingEnabled:bool
-CrosshairPos:Vector2D
-FixValueCrosshairPos:Vector2D
+AssignUIWidget():void
+UnassignUIWidget():void
+<event_listener>InputAxis():void
+<event_listener>ResetCombo():void
+<event_listener>InputAction():void
+<event_listener>OnKilled():void
+<event_listener>Pause():void
+<event_listener>WalletAuthentication():void
+<event_listener>TransferEther():void
+<event_listener>WidgetSelectorMenu():void
+<event>OnLanded():void
+<event>InitializeWeaponMenu:void
BP_Rifle
BP_GrenadeLauncher
BP_LauncherGrenade
BP_HealPowerUp
BP_SuperSpeedPowerUp
BP_SuperGunPowerUp
BP_LevelUpPowerUp
GameState
PlayerState
BP_GameState
BP_PlayerState

Weapon Factory Class on Integration Server



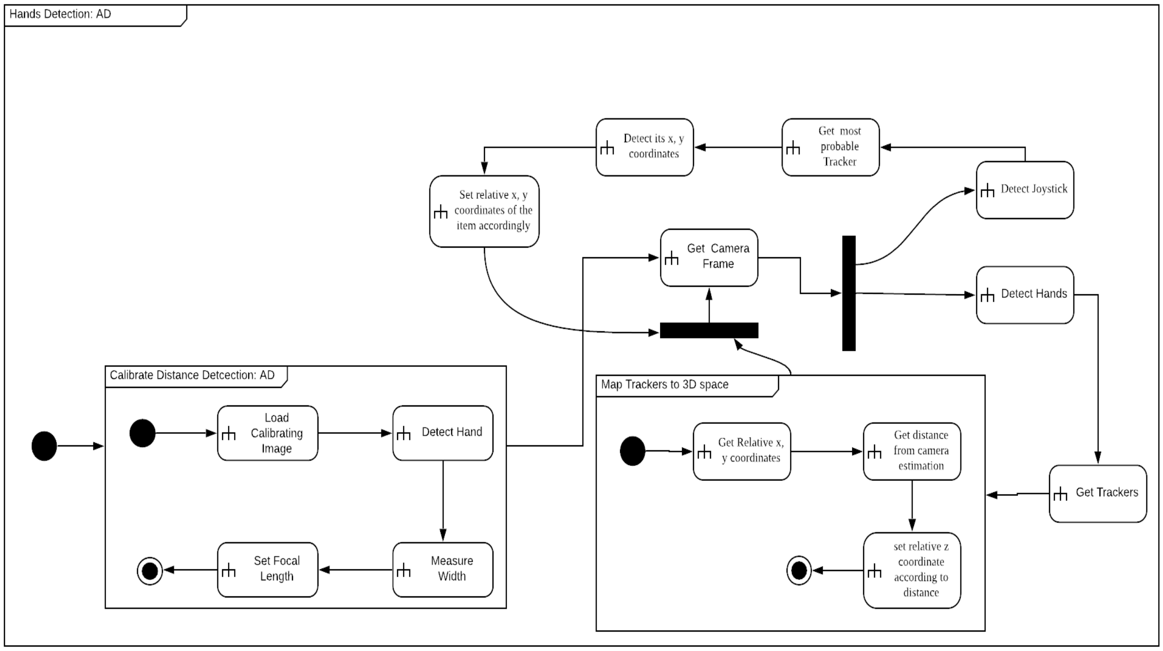
NewWeaponEventResponse	
log	Log
weaponId	BigInteger
name	String
wepType	BigInteger

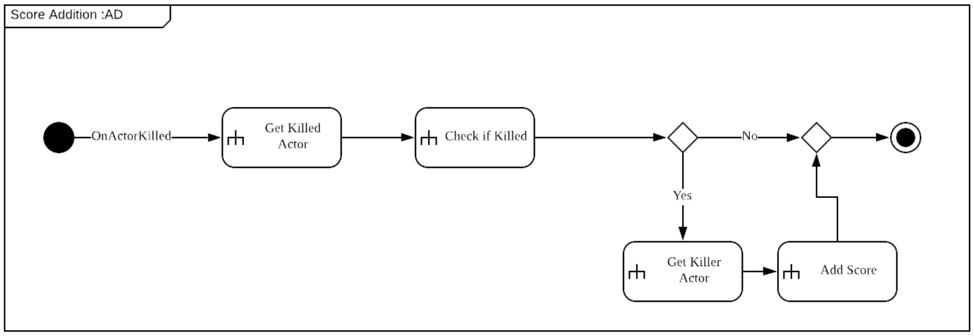
ApprovalEventResponse	
log	Log
_owner	String
_approved	String
_tokenId	BigInteger

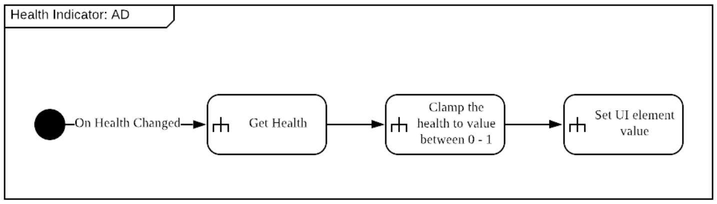
TransferEventResponse	
log	Log
_from	String
_to	String
_tokenId	BigInteger

WeaponFactory	
BINARY	String
FUNC_GETWEAPONBYOWNER	String
FUNC_TRANSFERFROM	String
FUNC_LEVELUP	String
FUNC_WEAPONTOOWNER	String
FUNC_CREATEWEAPON	String
FUNC__OWNEROF	String
FUNC_WEAPONS	String
FUNC_GETLEVELUPFEE	String
FUNC_BALANCEOF	String
FUNC_CHANGECREDS	String
FUNC_GETWEAPON	String
FUNC__NEWPLAYERCREATE	String
FUNC_BURNN	String
FUNC_CHANGENAME	String
NEWWEAPON_EVENT	Event
TRANSFER_EVENT	Event
APPROVAL_EVENT	Event
~~WeaponFactory(String, Web3j, Credentials, BigInteger, BigInteger)~~	
WeaponFactory(String, Web3j, Credentials, ContractGasProvider)	
~~WeaponFactory(String, Web3j, TransactionManager, BigInteger, BigInteger)~~	
WeaponFactory(String, Web3j, TransactionManager, ContractGasProvider)	
getWeaponByOwner(String)	RemoteCall<List>
transferFrom(String, BigInteger)	RemoteCall<TransactionReceipt>
levelUp(BigInteger)	RemoteCall<TransactionReceipt>
WeaponToOwner(BigInteger)	RemoteCall<String>
createWeapon(String, BigInteger)	RemoteCall<TransactionReceipt>
_ownerOf(BigInteger)	RemoteCall<String>
weapons(BigInteger)	RemoteCall<Tuple7<String, BigInteger, BigInteger, BigInteger, BigInteger, BigInteger, BigInteger>>
balanceOf(String)	RemoteCall<BigInteger>
changeCreds(BigInteger, BigInteger)	RemoteCall<TransactionReceipt>
getWeapon(BigInteger)	RemoteCall<Tuple7<String, BigInteger, BigInteger, BigInteger, BigInteger, BigInteger, BigInteger>>
_newPlayerCreate(String, BigInteger)	RemoteCall<TransactionReceipt>
burnn(BigInteger)	RemoteCall<TransactionReceipt>
changeName(BigInteger, String)	RemoteCall<TransactionReceipt>
getNewWeaponEvents(TransactionReceipt)	List<NewWeaponEventResponse>
newWeaponEventFlowable(EthFilter)	Flowable<NewWeaponEventResponse>
newWeaponEventFlowable(DefaultBlockParameter, DefaultBlockParameter)	Flowable<NewWeaponEventResponse>
getTransferEvents(TransactionReceipt)	List<TransferEventResponse>
transferEventFlowable(EthFilter)	Flowable<TransferEventResponse>
transferEventFlowable(DefaultBlockParameter, DefaultBlockParameter)	Flowable<TransferEventResponse>
getApprovalEvents(TransactionReceipt)	List<ApprovalEventResponse>
approvalEventFlowable(EthFilter)	Flowable<ApprovalEventResponse>
approvalEventFlowable(DefaultBlockParameter, DefaultBlockParameter)	Flowable<ApprovalEventResponse>
~~load(String, Web3j, Credentials, BigInteger, BigInteger)~~	WeaponFactory
~~load(String, Web3j, TransactionManager, BigInteger, BigInteger)~~	WeaponFactory
load(String, Web3j, Credentials, ContractGasProvider)	WeaponFactory
load(String, Web3j, TransactionManager, ContractGasProvider)	WeaponFactory
deploy(Web3j, Credentials, ContractGasProvider)	RemoteCall<WeaponFactory>
~~deploy(Web3j, Credentials, BigInteger, BigInteger)~~	RemoteCall<WeaponFactory>
deploy(Web3j, TransactionManager, ContractGasProvider)	RemoteCall<WeaponFactory>
~~deploy(Web3j, TransactionManager, BigInteger, BigInteger)~~	RemoteCall<WeaponFactory>
levelUpFee	RemoteCall<BigInteger>

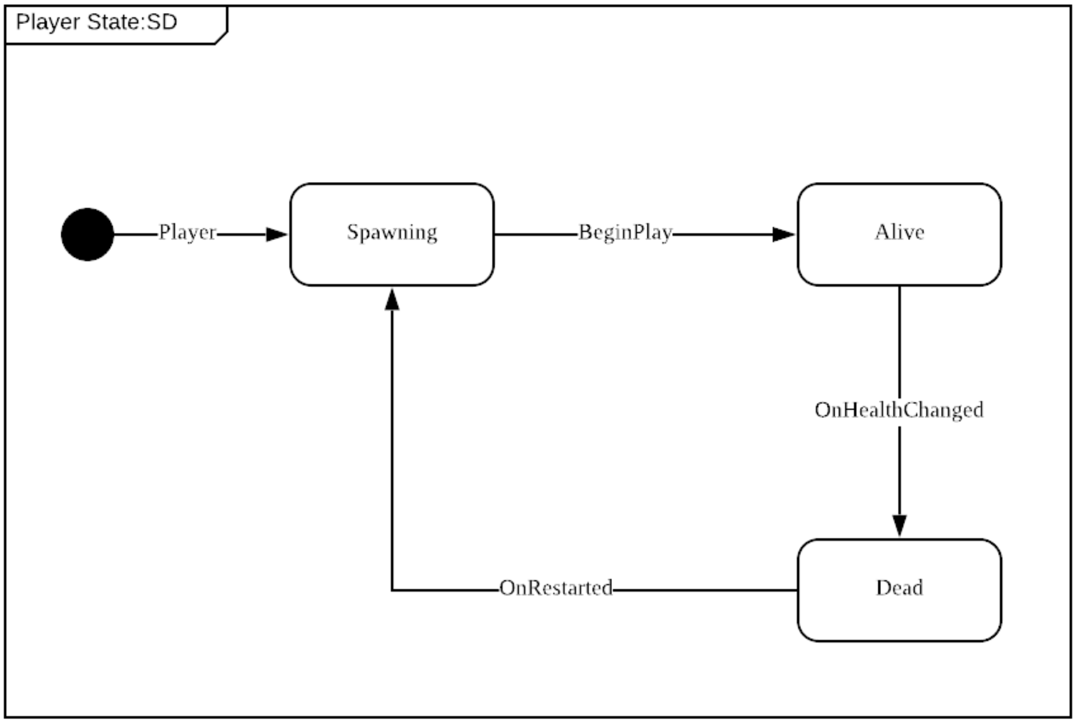
Activity Diagrams

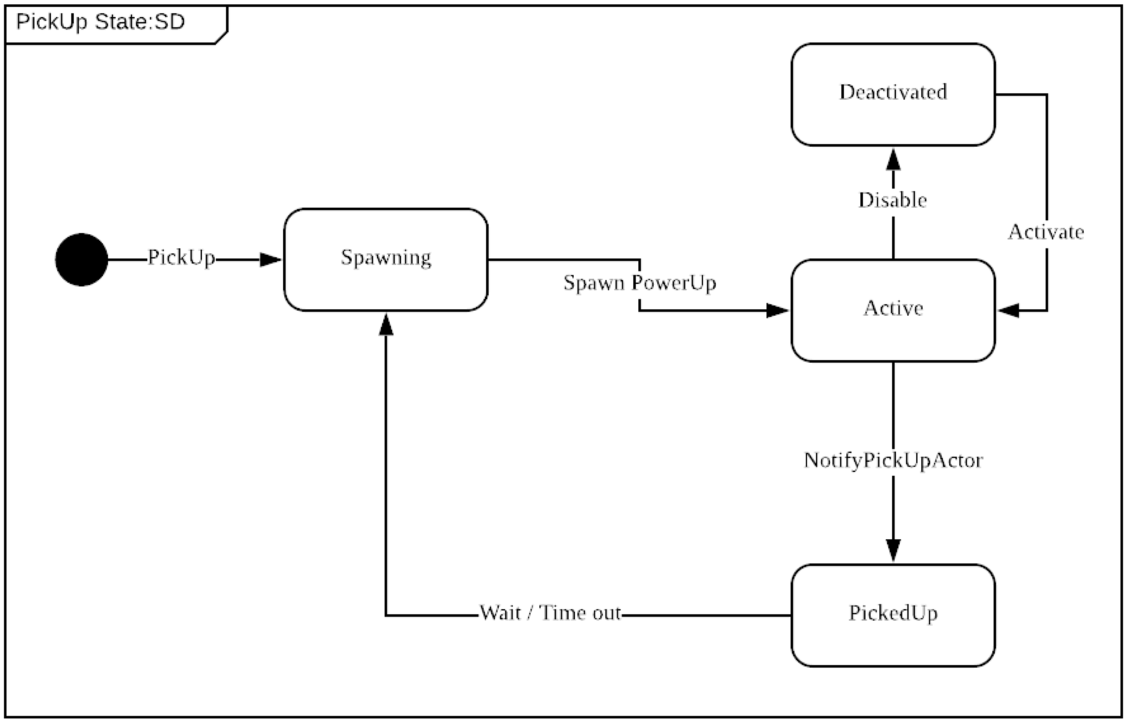

Hands Detection: AD
Detect its x, y coordinates
Get most probable Tracker
Detect Joystick
Set relative x, y coordinates of the item accordingly
Get Camera Frame
Detect Hands
Calibrate Distance Detcection: AD
Load Calibrating Image
Detect Hand
Measure Width
Set Focal Length
Map Trackers to 3D space
Get Relative x, y coordinates
Get distance from camera estimation
set relative z coordinate according to distance
Get Trackers





State Diagrams





8. System Development

The System whole development features the following technologies:

Game, VR and Tracking developed using Unreal Engine: UE4 Blueprint Language, C++, C#.

Blockchain Integration Server : Java Spring Framework.

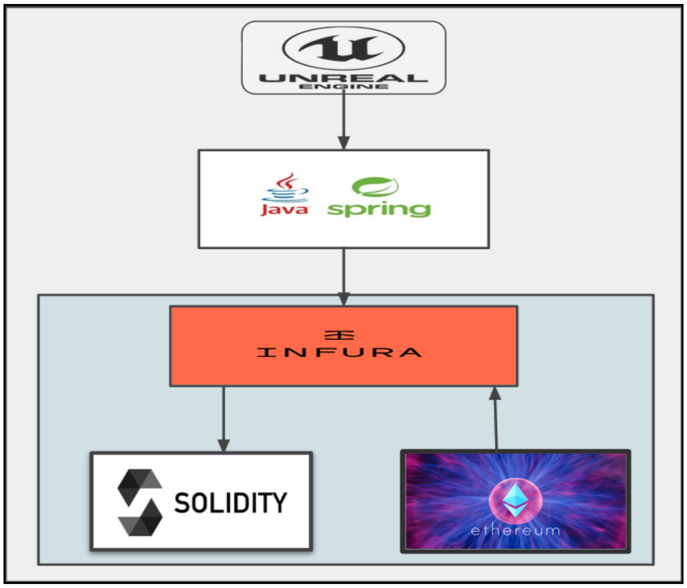
Blockchain Technology : Solidity Smart contracts, Ethereum Network.

Where the Coop shooter multiplayer game was originally developed using the Unreal Engine as a game development platform.

Blockchain Development using Ethereum:

The system can be described as having 3 layers:

- The game UI layer which interacts directly with the user developed inside the UE4.
- Java integration server layer to mediate between Ethereum endpoint and Game engine
- Backend layer composed of Infura - a hosted Ethereum node to interact directly with Ethereum blockchain.



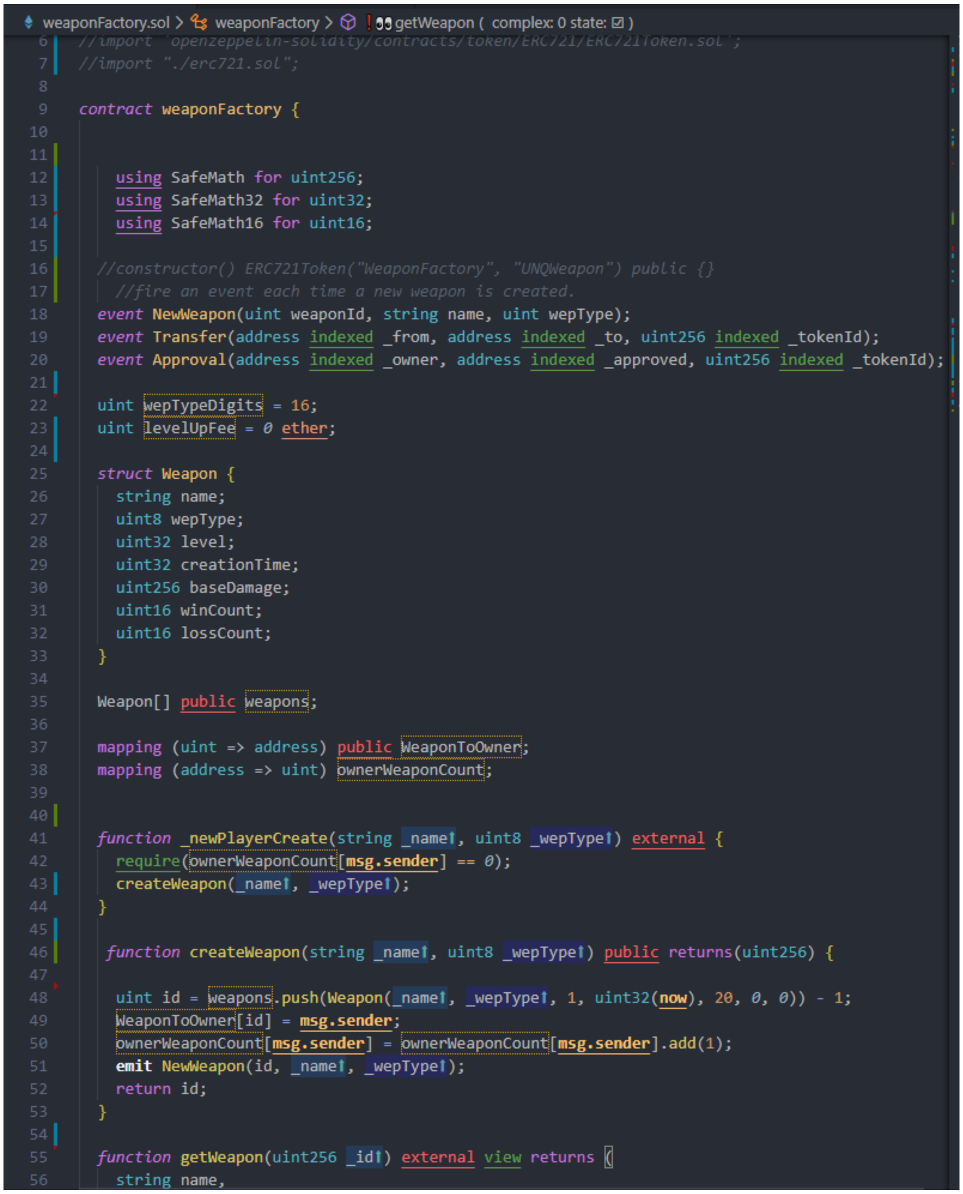
In order to set up our game backend on ethereum we need to develop smart contracts which require knowing a smart contract development language. We decided to use Solidity since it’s the primary smart contract development language on Ethereum networks. Smart contracts are tested, compiled and in the final stage deployed on ethereum blockchain where they live forever immune to altering or modifications. Usually deploying smart contracts on Ethereum mainnet requires spending Ether -Ethereum's native cryptocurrency- which can be bought for real money and is relatively expensive (1 Ether is approximately 130 USD). Since this project doesn't involve a budget for spending Ether we use the Ethereum test net Rinkeby which is free and simulates how the Ethereum mainnet works.

Once the contract is deployed on the ethereum network, the network gives the contract an address which is later used to interact with the methods inside this smart contract.

To be able to interact with the smart contract from the UE4 we needed to install an UE4 plugin called Ethere linker [6].

We also needed to develop a customized Integration server for our game, written in Java, which interacts directly with Ethereum blockchain endpoint. The integration server uses Ethereum's Web3j library to compile the solidity smart contracts and generate a Java wrapper for the methods inside it.

The UE4 Plugin sends requests to the integration server and gets responses from it. The interaction with the integration server is via JSON requests via HTTPS. Currently, integration server is using Infura endpoint to interact with Ethereum blockchain.


weaponFactory.sol > weaponFactory > getWeapon (complex: 0 state: ☑)

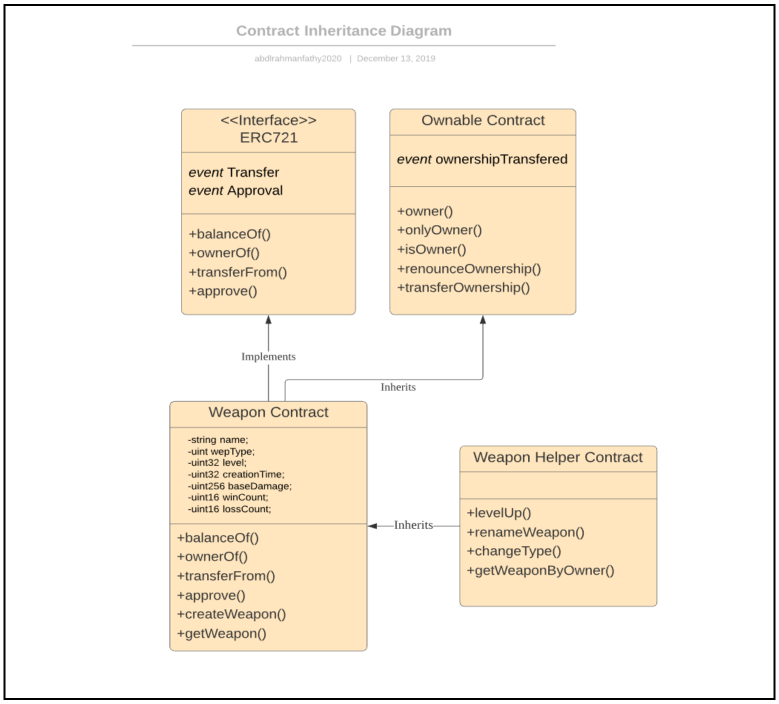
```solidity
//import "openzeppelin-solidity/contracts/token/ERC721/ERC721Token.sol";
//import "./erc721.sol";

contract weaponFactory {


    using SafeMath for uint256;
    using SafeMath32 for uint32;
    using SafeMath16 for uint16;

  //constructor() ERC721Token("WeaponFactory", "UNQWeapon") public {}
    //fire an event each time a new weapon is created.
  event NewWeapon(uint weaponId, string name, uint wepType);
  event Transfer(address indexed _from, address indexed _to, uint256 indexed _tokenId);
  event Approval(address indexed _owner, address indexed _approved, uint256 indexed _tokenId);

  uint wepTypeDigits = 16;
  uint levelUpFee = 0 ether;

  struct Weapon {
    string name;
    uint8 wepType;
    uint32 level;
    uint32 creationTime;
    uint256 baseDamage;
    uint16 winCount;
    uint16 lossCount;
  }

  Weapon[] public weapons;

  mapping (uint => address) public WeaponToOwner;
  mapping (address => uint) ownerWeaponCount;


  function _newPlayerCreate(string _name, uint8 _wepType) external {
    require(ownerWeaponCount[msg.sender] == 0);
    createWeapon(_name, _wepType);
  }

   function createWeapon(string _name, uint8 _wepType) public returns(uint256) {

    uint id = weapons.push(Weapon(_name, _wepType, 1, uint32(now), 20, 0, 0)) - 1;
    WeaponToOwner[id] = msg.sender;
    ownerWeaponCount[msg.sender] = ownerWeaponCount[msg.sender].add(1);
    emit NewWeapon(id, _name, _wepType);
    return id;
  }

  function getWeapon(uint256 _id) external view returns (
    string name,
```

Figure: Snapshot of the WeaponFactory Smart contract

Creating unique and scarce game assets using ERC721 and NFTs: In order for our game assets to be unique and scarce game contracts must comply with the Ethereum ERC721 Tokens Standard for creating Non-fungible tokens.

Contracts also need to inherit from Ethereum's "Ownable" interface, to ensure exclusive rights to item owners and to give the contract owner administrative permissions over some functions.

Interaction with blockchain requires authenticated wallet address and signature. Some contract functions require certain criteria to be met before execution.



9. System Evaluation

Whitebox Testing

Test Case ID	Test Case Scenario	Input Values	Expected Results	Actual Results	Pass/Fail
TC-01	Adding VR support with android	Building for android X64 architecture	Build running on android	App crashes on open	Fail
TC-02	Adding VR support with android	Building for android armv7 architecture	Build running on android	Build running on android	Pass
TC-03	Adding VR support with iOS	Building for iOS 12	Build running on iOS	Build running on iOS	Pass
TC-04	Adding TensorFlow as an external library	Testing with a sample image detection code	Working on PC	Working on PC	Pass
TC-05	Adding OpenCV as an external library	Testing with a sample image detection code	Working on PC	Working on PC	Pass
TC-06	Adding TensorFlow as an external library	Testing with a sample image detection code	Working on android seamlessly	Library failed to integrate with UE4 on android	Fail

TC-07	Adding OpenCV as an external library	Testing with a sample image detection code	Working on android seamlessly	Library failed to integrate with UE4 on android	Fail
TC-08	Testing Skin Isolation algorithm using UE4 shaders	Getting images from PC camera	Being able to process without frame loss	Being able to process without frame loss	Pass
TC-09	Getting access to mobile camera directly	Using image acquisition from system library	Be able to access mobile camera	Error incompatible command	Fail
TC-10	Getting access to mobile camera using android library	Using image acquisition from android library	Be able to access android camera	Be able to access android camera	Pass
TC-11	Moving a tracker using detected skin	Using PC camera and skin isolation function to track position on screen	Be able to locate tracker on screen	Be able to locate tracker on screen	Pass
TC-12	Building HandTracking Component to Android	Using android camera and HandTracking Component to move crosshair	Working properly	Working properly	Pass
TC-13	Adding Projectile	Using the SWeapon	Shoot	Shoot	Pass

	Launcher weapon with game	class to create new shooting mechanic	projectile	projectile	
TC-14	Firing using updated crosshair position	Using crosshair position to determine shot direction	Shoot to crosshair position	Shoot to crosshair position	Pass
TC-15	Building game mechanics to android	Using premade textures and level assets	Working properly	Some textures missing	Fail
TC-15	Building game mechanics to android	Using mobile textures and materials	Working properly	Working properly	Pass
TC-16	Using Multiplayer listener server	Using a listener server to test multiplayer gameplay	Shooting,movement and health replicated.	Shooting,movement and health replicated.	Pass
TC-16	Using Multiplayer dedicated server	Using a dedicated server to test multiplayer gameplay	Shooting,movement and health replicated.	Client players killed on game start	Fail

Integration Testing

Test Case ID	Test Case Scenario	Input Values	Expected Results	Actual Results	Pass/Fail

TC-01	Integrating VR with the shooting game mechanics	Building for android armv7 architecture	Build running on android	Build running on android	Pass
TC-02	Integrating HandTracking Component with Game Mechanics	Using PC camera and HandTracking Component to move crosshair	Working properly	Working properly	Pass
TC-03	Integrating HandTracking with VR and Game Mechanics	-	Working properly	Working properly	Pass
TC-03	Integrating HandTracking with VR and Game Mechanics with a listener server	-	Working properly	Working properly	Pass

Functionality testing

Test Case ID	**Test Scenario**	**Test Steps**	**Test Data**	**Expected Results**	**Actual Results**	**Pass/Fail**
TC-001	Retrieving the asset	- Call the getWeapon	Weapon Id	Returns weapon's:	Returns weapon's:	pass

	attributes from the blockchain inside the game	function from the testing environment		Name, id, level, damage and creation time	Name, id, level, damage and creation time	
TC-002	Creating a new asset represented by NFT on the blockchain inside the game	- Performing the create weapon function from the testing environment	Weapon name, weapon Type	Returns new weapon Id	Returns new weapon Id	Pass

Test Case ID	**Test Scenario**	**Test Steps**	**Test Data**	**Expected Results**	**Actual Results**	**Pass/Fail**
TC-003	Leveling up a weapon during game play	- Player picks up a buff while playing - Action is triggered on the blockchain	Weapon Id	Weapon level increases by 1 level	Weapon level increases by 1 level	Pass

TC-004	Change the Weapon name from inside the game UI	- Type in the new weapon name - Press confirm transaction	Weapon Id, New weapon name	Weapon name changes on the blockchain	Weapon name changes on the blockchain	Pass

Test Case ID	**Test Scenario**	**Test Steps**	**Test Data**	**Expected Results**	**Actual Results**	**Pass/Fail**
TC-005	Transfering a weapon to a new owner form game UI	- Enter Receiver address and weapon Id - Confirm transaction	Receiver address, Weapon Id	Asset is transferred to the new player	Asset is transferred to the new player	Pass
TC-006	Deleting a weapon from the player’s wallet	- Enter the weapon ID - Press confirm Deletion	Weapon Id	Weapon is deleted from the player’s wallet	Weapon is deleted from the player’s wallet	Pass

10. Conclusion and Commercialization

This project was intended for the use of proving the ability of using the blockchain technology for basing game assets on, and to persuade developers to leverage the power of mobile devices for the development of a platform that cost effectively can replace the high cost of high-end trackers for expensive VR headsets and the overhead of using such equipment, both using a mobile multiplayer shooter game for the sake of demonstration, using Unreal Engine 4 as a development tool.

This document starts with the description of the proposed problems' statements, and proceeds by giving the research and data needed to prove the need of finding solutions to these problems, and then continues to provide a possible solution to the suggested problem, along with all the design and architecture needed to develop such a system.

In the previous parts we discussed the developed system as a proof of concept, and explained in detail the process of development, specifying the classes, modules and libraries used during implementation, and then explaining the test cases that the system has been through during development.

In final analysis, we have developed a system that can fulfil the objectives presented, where it can:

- Detect hand and trackers using a real-time hand tracking module on mobile devices, without affecting gameplay.
- Creating a VR with tracker aiming mechanics using low-cost equipment.
- Storing game assets on the ethereum blockchain, where they can be edited, traded, and owned by the individual players.
- Retrieving the assets stored on the blockchain with the minimum overhead required, while running on a mobile device.

Provided the later, this project has fulfilled its objectives of having a proof of concept for the problem statements proposed earlier.

11. References

[1] (2019). Retrieved 21 August 2019, from https://www.researchgate.net/profile/Richard_Twesige/publication/270287317_Bitcoin_A_simple_explanation_of_Bitcoin_and_Block_Chain_technology_JANUARY_2015_RICHARD_LEE_TWESIGE/links/54a7836f0cf267bdb90a0ee6/Bitcoin-A-simple-explanation-of-Bitcoin-and-Block-Chain-technology-JANUARY-2015-RICHARD-LEE-TWESIGE.pdf

[2] Ranked list of blockchain games | DappRadar. (2019). Retrieved 21 August 2019, from https://dappradar.com/rankings/category/games

[3] Hudzilin, A. (2019). The State of the Blockchain Gaming Industry. Retrieved 21 August 2019, from https://hackernoon.com/the-state-of-the-blockchain-gaming-industry-3d3445dc2ec

[4] Genesis Content: Funding the Future of Blockchain Gaming. (2019). Retrieved 21 August 2019, from https://medium.com/decentraland/genesis-content-funding-the-future-of-blockchain-gaming-a5cb55dca8c5

[5] Blockchain Games - Can Blockchain Technology be a Game Mechanic? - Extra Credits. (2019). Retrieved 22 August 2019, from https://www.youtube.com/watch?v=ywvTIM_eOVI

[6] https://bitbucket.org/kelheor/etherlinker-for-ue4/src/master/