

NFT FOR GAME DEVELOPERS

A Thesis
presented to
the Faculty of Natural and Applied Sciences
at Notre Dame University-Louaize

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in Computer Sciences

by

CHARBEL SALIBA

07 2022

© COPYRIGHT

By

Charbel Saliba

2022

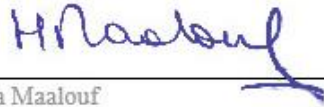
All Rights Reserved

Notre Dame University - Louaize
Faculty of Natural and Applied Sciences
Department of Computer Sciences

We hereby approve the thesis of

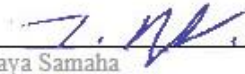
Charbel Saliba

Candidate for the degree of Master of Science in Computer Sciences



Dr. Hoda Maalouf

Supervisor, Chair



Dr. Maya Samaha

Committee Member

Declaration

I declare that this thesis has been composed solely by myself and that it has not been submitted, in whole or in part, in any previous application for a degree. Except where states otherwise by reference or acknowledgment, the work presented is entirely my own.

Acknowledgments

Thank you!

Table of Contents

Acknowledgments	v
Table of Contents	vi
List of Figures	viii
List of Abbreviations	x
Abstract.....	xi

Chapter 1: Introduction and Problem Definition..... 1

1.1 What is NFT?	1
1.1.1 Video Games and NFT	3
1.2 Problem Definition	5
1.3 Research Objectives	5
1.4 Approach and Main Results	6
1.5 Thesis Organization.....	6

Chapter 2: Background and Motivation 7

2.1 Definition of the Basic Concepts.....	7
2.1.1 Emergence Of Blockchain	7
2.1.2 Consensus Model and Mining	9
2.1.3 Cryptocurrency	10
2.1.4 Ethereum and Smart Contracts	12
2.1.5 Gas Fees.....	13
2.1.6 NFT	14
2.1.7 ERC-721	15
2.2 Analysis of NFT Integration in Video Games.....	15
2.2.1 Decentralized Ownership.....	16
2.2.2 Interoperability Of Game Items.....	18
2.2.3 Play-to-Earn	20
2.3 Descriptions of the Used Software and Algorithms	21
2.4 Previous Work in the Subject	21
2.5 Research Motivation.....	21

Chapter 3: Original Work 23

3.1 NFT-Friendly Design	23
3.1.1 Crypto Run.....	24
3.1.2 Crypto Forgers	25

3.2 NFT Integration	26
3.2.1 Ganache	27
3.2.2 Smart Contract	27
3.2.3 Creating and Loading Tokens	30
Chapter 4: Conclusion	34
4.1 Summary of the Main Results	34
4.1.1 The process	34
4.1.2 The challenges	34
4.2 Main Contributions of the Thesis	36
4.3 Possible Extensions and Future Work	36
Bibliography	38

List of Figures

Figure 1.1- Kevin McCoy, Quantum	2
Figure 1.2 - Beeple, Everyday: The First 5000 Days	2
Figure 1.3 – CryptoKitties with Different Genes	4
Figure 1.4 - Axie Infinity Gameplay	4
Figure 2.1 - Blockchain Representation	8
Figure 2.2 - Types of Blockchains	9
Figure 2.3 - Proof of Work	10
Figure 2.4 - A Mining Farm	11
Figure 2.5 - Transaction Procedure	12
Figure 2.6 – Ethereum Average Transaction Fee Between October 2019 and July 2022	13
Figure 2.7 - 3D Leather Bag	16
Figure 2.8 - Marvel-Based Characters in Fortnite’s Item Shop	19
Figure 3.1 - Crypto Run Gameplay Footage	24
Figure 3.2 - Crypto Run Main Menu UI	25
Figure 3.3 - Crypto Forgers Gameplay UI	26
Figure 3.4 - Ganache	27
Figure 3.5 - ERC-721 Smart Contract	28
Figure 3.6 - ItemToken Contract Migration	29
Figure 3.7 - Contract Deployment Fees	29
Figure 3.8 - Contract Connection Code Snippet	30
Figure 3.9 - NFT Generation	31
Figure 3.10 - Metamask Transaction Prompt	31

Figure 3.11 - Loading Tokens Ids	32
Figure 3.12 - Loading Token Data.....	32

List of Abbreviations

- **NFT:** Non-Fungible Token
- **dApps:** Decentralized Applications
- **VC:** Venture Capitalist
- **SLP:** Smooth Love Potions

Abstract

This research aims to validate the viability of implementing NFT mechanics in a video game. It discusses the blockchain technology's limitations, as well as the many restrictions that NFT forces on game developers. Be it their immutable nature, their high minting cost, or their incapacity of holding big data. It also highlights the restrictions that prevent the interoperability of game assets, and finally, the ludicrous Play-to-earn business model.

Two games are also developed, Crypto Forgers and Crypto Run, to further validate the findings presented by this research. While these two games are simple in gameplay terms, they excellently showcase the inadequate integration of NFT game features.

Chapter 1: Introduction and Problem Definition

1.1 What is NFT?

Minting is the process of creating a Non-Fungible Token, or NFT, a one-of-a-kind, digital asset that resides on the Ethereum blockchain, and represents real world objects, like music, art, in-game items, videos, etc. The item itself is not necessarily unique, as anyone can just save a copy of an image on the internet for themselves, however, the transaction on the blockchain's ledger guarantees the current owner, the original creator, as well as the unique history of ownership of the item. For example, let us assume that the "Mona Lisa" was an NFT. Anyone could have a copy of the original image on their device (seeing that it is not a painting), or mint their own token, and offer it on a marketplace, like OpenSea, however, there would only be one authentic token, created by Leonardo da Vinci, and could be easily identifiable given the history of ownership, as he would be its first owner.

Although "Quantum", the first-ever NFT, minted by artist Kevin McCoy, was created in 2014, it wasn't until 2021 that the topic would gain an immense surge in popularity among avid investors, entrepreneurs, online content creators, digital artists, crypto enthusiast, and many more.

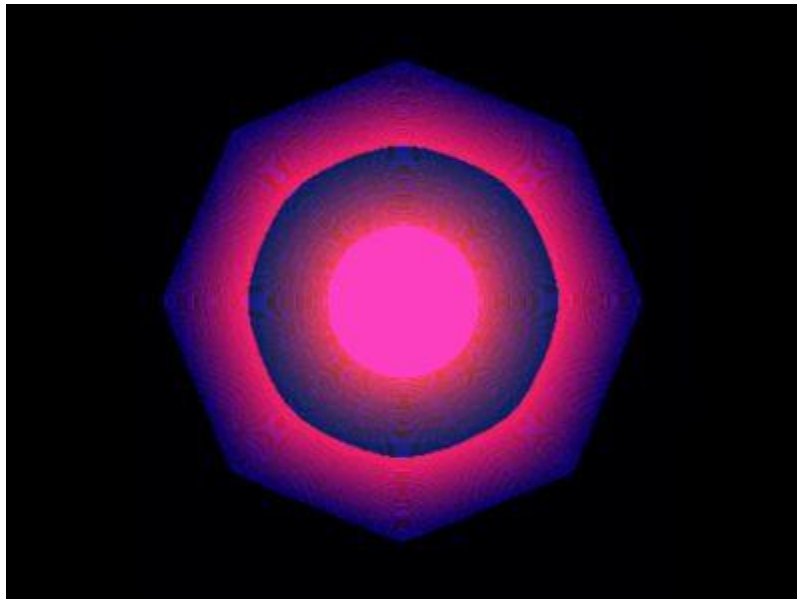


Figure 1.1- Kevin McCoy, Quantum

In the past year, sales of NFTs reached around \$17.6 billion, compared to just over \$82 million the year before, an outstanding increase of 21,000%. “Everydays: The First 5000 Days”, created, and, minted by artist “Beeple”, alone was traded for the hefty sum of \$69,346,250. Purchasing an NFT became more of an investment, a speculation that the price of it will go up, given the community’s engagement in the project, or simply, bragging rights, as an additional piece to someone’s portfolio. The value did not originate from the art itself.

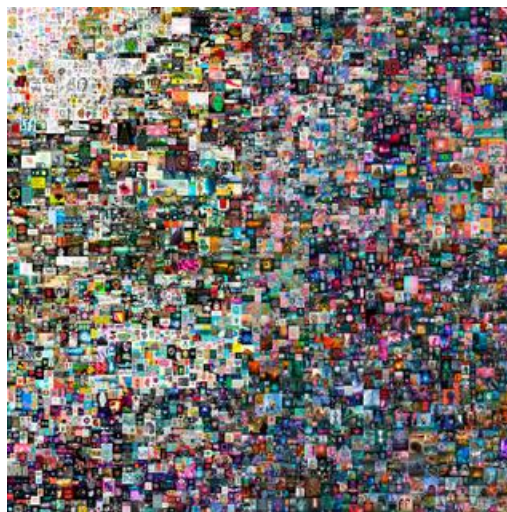


Figure 1.2 - Beeple, Everydays: The First 5000 Days

With NFT being the biggest trend of 2021, tech enthusiasts, Venture Capitals, entrepreneurs, and developers have shown keen interest in its underlying technology, Blockchain. It started with the creation of bitcoin, in 2008. Satoshi Nakamoto, pseudonym of the writer of the bitcoin whitepaper, detailed an innovation that would allow a peer-to-peer cash exchange system, outside of the governance of centralized entities, like banks, or governments. Simply put, it is a decentralized, distributed database, formed of two data structures: blocks and transactions. A transaction represents an exchange between two wallets, such as, Person A pays 23 bitcoins to person B. A block encompasses a list of transactions, and has a certain size limit. Once a block is full, a new block is created, and is referenced by its predecessor, hence, a chain of blocks. By design, these blocks are public and immutable. The information of the transactions is available to all, and unalterable, which creates, coupled with a Proof of Work consensus protocol, a layer of trust between the user and the system.

Over the last decade, blockchain technology has had an influence on various industries, such as logistics, finance, health-care, entertainment, etc. And being significantly sensitive to technological novelty, a new business model is introduced to the Game industry, play-to-earn, that builds upon blockchains and NFTs.

1.1.1 Video Games and NFT

CryptoKitties is the first widely recognized Ethereum-based blockchain game. Players can purchase, sell, and breed virtual cats. Every CryptoKitty is an NFT, it is unique and owned by a player. To start the game, players have to purchase an initial CryptoKitty from the marketplace, which price ranges from one U.S. dollar, to over half a million. Cats are made up of traits and genes categorized in multiple rarities, which creates an artificial scarcity of a mix

of traits, that gives tokens their value. The rarer and more coveted a gene is, the higher the price of cats possessing it.



Figure 1.3 – CryptoKitties with Different Genes

Another popular blockchain-based game is Axie Infinity. A card game that revolves around collecting monsters, known as axies, and battle against online opponents. It also offers its players the ability to trade and breed new axies. Every axie is an NFT. Ranked battle mode costs players energy points that replenish daily, and upon victory, and achieving higher ranks, grants the player a certain amount of Smooth Love Potions (SLP). These potions, that enable the players to breed new axies, is also an NFT, and is the main currency exchanged on the market. Players eager to produce a revenue will prioritize farming and selling these potions.



Figure 1.4 - Axie Infinity Gameplay

These games, among multiple others, such as Gods Unchained, Splinterlands, The Sandbox, and more, present a new business model to the game industry, play-to-earn. The main reason players engage in these games is for financial compensation.

1.2 Problem Definition

NFT has become a major hype surrounding the game industry, every player wants to personally own their items, and receive a return on investment for their time playing the game. While some games have implemented these models, no one is questioning the validity of the product itself. When analyzed thoroughly, Play-to-Earn games are nothing more than a Greater Fools Scam. Older players can cash out from the investment of new players into the game, and, at some point, the game will run out of new players.

True ownership of game items, interoperability of game assets, and Play-to-Earn. This thesis analyzes the impact of integrating NFTs on the game development industry, from a technical point of view.

1.3 Research Objectives

True ownership of player items, interoperability of game assets, and a novel business model, play-to-earn. This research aims to discover the feasibility and complexity of integrating NFTs, and blockchain-based features, in modern video games. Additionally, it will also discuss the impact of the nature of these mechanics on the design of game worlds.

1.4 Approach and Main Results

Two games were developed in order to assess the different mechanics built upon NFT technology. The current tech stack of blockchain, coupled with the limitations of the system itself, has shown that forcing NFTs into game ecosystems disrupts the flow of the game. Because developers have to program game functionalities at a minute level, any outsider data would crash at runtime. Additionally, with the current constraints of blockchain technology, and smart contracts, player ownership of game assets is, at best, illusory. What players actually own, are links, or data structures, that make no sense outside of the game's environment. And the idea, that owning a game item NFT, will preserve its value outside of the game, even if the game server's shutdown, is completely erroneous. The value of an NFT is driven by the demand of the market, a token that cannot be linked to an active game, will simply become useless.

1.5 Thesis Organization

This thesis starts by discussing the current state of blockchain technology and game development in chapter 1. In chapter 2, we outline the technical components of the blockchain, and cryptocurrency. Followed by an analysis of NFT integration in video games, with an overview of the envisioned mechanism achieved. In chapter 3, two Ethereum-based blockchain games are developed, that help assess the impact of incorporating NFTs on the game development process. Finally, in chapter 4, a conclusion is built based on the analysis of chapter 2, and the findings of chapter 3, and opens the door on new research potential.

Chapter 2: Background and Motivation

In order to highlight the impact of integrating Non-Fungible Tokens on universal game development methods, this section makes a point to define the technical benefits and drawbacks of the underlying technology, blockchain, and its implementations, and, gives a general overview of the different elements of the game design process.

2.1 Definition of the Basic Concepts

2.1.1 Emergence Of Blockchain

Blockchain technology was first drafted in 1991, when two researchers, Stuart Haber and W. Scott Stornetta, wanted to implement a system that prevented tampering with documents' timestamps. However, it wasn't until 2008 that it would get its first real-world application with the launch of the first decentralized cryptocurrency, Satoshi Nakamoto's Bitcoin.

A blockchain is a distributed database shared among multiple nodes of a computer network. The information stored mostly represents transactions; for example, person A sends 100 tokens to person B. The key difference, however, between a blockchain and a typical database, is that a blockchain is an append-only ledger. The data stored is immutable. Nodes can only add new transactions. Old transactions cannot be altered, or, deleted. Transactions are gathered in groups, known as blocks. Blocks are of limited capacity, and when filled, are closed, and linked through hashes, to their predecessor, forming a chain of blocks. The first block is notably called "Genesis Block", and is not linked to any previous group.

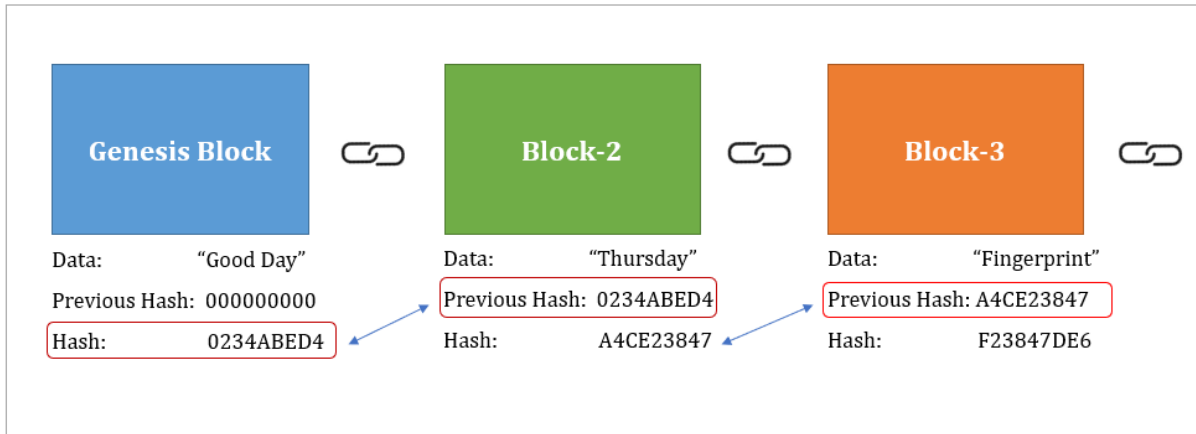


Figure 2.1 - Blockchain Representation

According to Puthal et al., we can distinguish four types of blockchain systems. Public blockchains, also known as “permission-less” blockchains, allow anyone to read existing transactions, add new ones, and append blocks to the chain. These blockchains are fully decentralized, and all participating nodes have a full copy of the data. Of course, a consensus algorithm, such as Bitcoin’s proof-of-work model, is employed to validate and authorize new blocks. Bitcoin, Ethereum, Cardano, Litecoin, etc. are amongst the most popular public blockchains. Private blockchains, or “permissioned” blockchains, are managed by a single organization. This central authority decides who can be a node, or a user, in the network. Because access is restricted, these blockchains are only partially decentralized. Consortium blockchains are permissioned blockchains governed by a group of organizations. Finally, hybrid blockchains are controlled by a single entity, but offer some permission-less services. One of the main purposes of implementing game assets as NFTs, is to create true ownership of an in-game item, for the user, independent of the game company’s servers, and as such, the blockchain ought to be public.

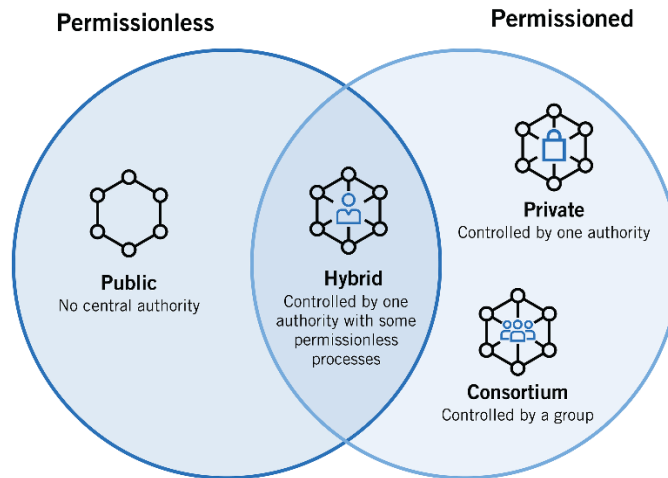


Figure 2.2 - Types of Blockchains

It is important to draw a clear distinction between a node and a user on the network. A user can only initiate new transactions, but it is up to the node to validate and append the new record on the active block. Users can opt to become a node, or member, by hosting the full blockchain locally, receive users' transactions, and create new blocks. This process is known as "mining".

2.1.2 Consensus Model and Mining

On a public blockchain, any user can become a node. Therefore, the system is susceptible to attacks that create fraudulent blocks. Consensus models are techniques that prevent spamming blocks into the chain. The Bitcoin blockchain implements the Proof of Work model. Proof of Work was first drafted as a counter-measure for e-mail spammers. The system would require the machine to solve a mathematical puzzle, that would take a couple of minutes, before sending an email.

To append a new block on the chain, "miners" have to sign the block with a cryptographic signature based on the recorded transactions and a generated nonce, using the SHA-256 hashing

algorithm. To make the process more computationally complex, the hash of the block has to start with an ever-increasing number of leading zeros, currently being 17. Finding a compatible nonce, usually a random integer, takes currently on average around 10 minutes. Thus, finding a valid signature, proves the work put in by the node, and therefore, guarantees the validity of the block.

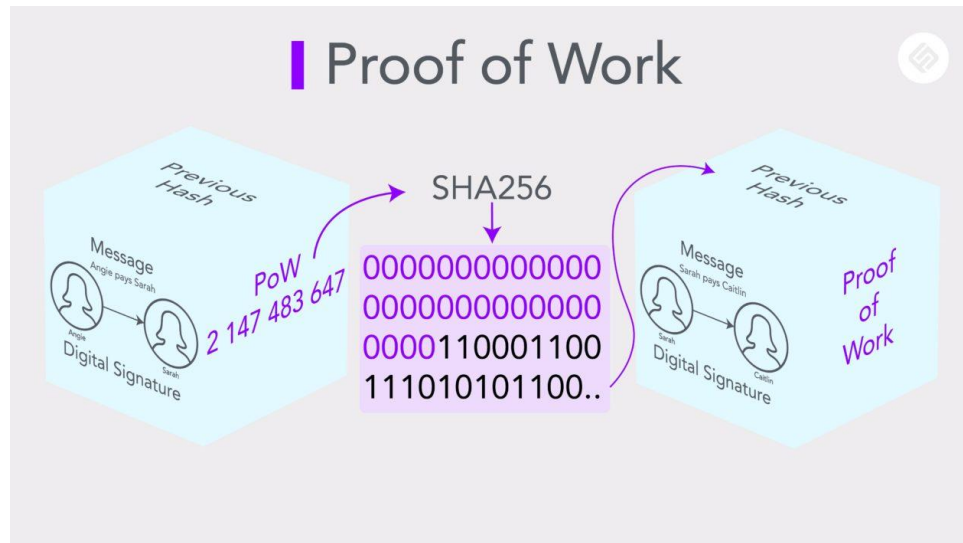


Figure 2.3 - Proof of Work

2.1.3 Cryptocurrency

Mining new blocks is computationally expensive; however, miners have found that simply employing more machines, leads to an increased chance of finding a valid signature faster. Nodes have competed to create what is known as a mining farm, utilizing hundreds of graphics processing units, to improve their rate of success. This has led to an insane amount of energy used, where globally, bitcoin rigs are using around an estimated 150 terawatt-hours of electricity annually — more than the entire country of Argentina, with a population of 45 million.

So mining is costly, therefore, miners expect to be properly compensated. This is the role of the cryptocurrency. When a node successfully signs, validates, and appends a new block to the bitcoin chain, they are rewarded with one bitcoin. This is the incentive. The reason users opt to become active nodes on a blockchain network.



Figure 2.4 - A Mining Farm

Thanks to cryptography, cryptocurrency, common implementation of blockchain, is a secure medium of exchange. The decentralized nature of the network gives each peer an equal position, since they each have a replica of the ledger, while hashing algorithms validate and authorize ongoing transactions. The private/public key algorithm is used. Every user on the blockchain possesses a private key, that only they know, and a public key that everyone else has access to. When creating a new transaction, the record is signed by the sender, using their private key. The public key, then, enables the nodes on the server to validate the authenticity of the exchange, before appending it to the current block.

For example, assume person A sends 10 coins to person B. A cryptographic signature, hashed using person A's private key, and, the data, such as, the amount exchanged, is sent along the transaction. If the network's nodes cannot replicate the message's content using person A's

public key, this entails that the transaction is fraudulent, and it will not be appended to the ledger.

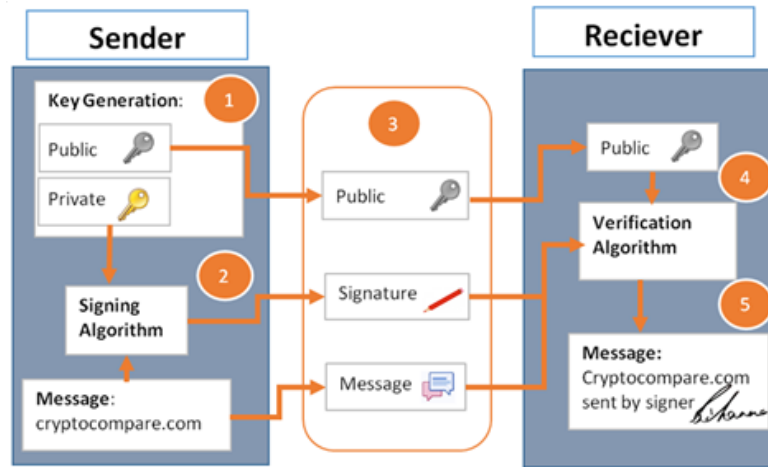


Figure 2.5 - Transaction Procedure

2.1.4 Ethereum and Smart Contracts

Rather than becoming the new decentralized digital currency, that people would actually use to buy stuff, Bitcoin had become an unwieldy speculative financial instrument, too slow and expensive to use for anything. In practice, you couldn't do anything but bet on it. Additionally, the capabilities of the system itself was limited to being a repository of transactions between accounts.

In 2014, crypto enthusiast Vitalik Buterin and his peers launched Ethereum. A competing cryptocurrency that boasted lower fees, faster transaction times, a reduced electrical footprint, and, most notably, a sophisticated processing functionality. Buterin aimed to make Ethereum programmable, so developers can build and deploy decentralized crypto-based apps, or dApps, on the network. In addition to tracking Ether coins, the ledger would also be able to track arbitrary blocks of data. As long as they were compatible with the structure of the Ethereum network, these blocks of data could also be functional programs, or, Smart Contracts. While

Bitcoin is only a payment network, Ethereum is more like a marketplace of financial services, games, social networks and other apps.

A Smart contract is a program with the terms of agreement directly written into lines of code, that executes when predefined conditions are met. Because smart contracts reside on the blockchain, the code contained is immutable, which entails absolute rules, and full transparency. All parties of the contract can be immediately certain of the outcome, without the need of a central authority, legal system, or external enforcement mechanisms.

2.1.5 Gas Fees

Ethereum currently uses the Proof-of-Work consensus model. Therefore, miner nodes ought to be rewarded with Ether coins for successfully validating a new block. However, unlike the Bitcoin blockchain, Ethereum introduces transaction fees on the users, which are added to the 2 ether coins reward.

Gas refers to the unit that measures the computational effort required to execute specific operations on the Ethereum network. Since each Ethereum transaction requires computational resources to execute, each transaction requires a fee. Gas refers to the fee required to conduct a transaction on Ethereum successfully. Gas prices are denoted in gwei, a denomination of Ether, with $1 \text{ gwei} = 10^{-9} \text{ ether}$.

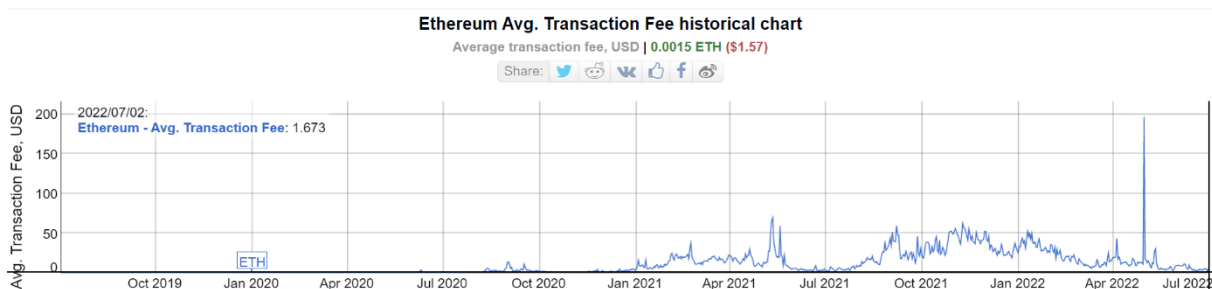


Figure 2.6 – Ethereum Average Transaction Fee Between October 2019 and July 2022

Smart contracts also come with their set of fees. According to the whitepaper, Ethereum charges 20,000 gwei per 256 bits of data. The more the size of your contract, the more will be the gas price associated with its creation. This is separate from the transaction fee required to deploy the smart contract on the chain. Additionally, the allocation of memory for a contract execution costs 200 gwei per byte. So, on average, deploying a simple contract can cost anywhere between \$400 and \$2000.

2.1.6 NFT

Enabling tokens to hold blocks of data gave birth to the concept of Non-Fungible Token on the Ethereum blockchain. NFTs are cryptographic assets with a unique identifier, and metadata that distinguish them from each other. As opposed to cryptocurrency, described as fungible tokens that can be traded at equivalency, NFTs each possess a distinct value. These tokens merge both the lossless replication attribute of digital goods, and the uniqueness of physical objects. While two tokens could contain identical data, they are still distinct instances. An NFT can only have one owner at a time, which is managed by the chain. Owners can choose to gather NFTs as a form of digital collectibles, or sell them on marketplaces, in exchange for ether coins. Think of it as how books exist in millions of copies, however, each copy is a unique instance of the book. If someone were to sell a copy owned and signed by the author, then the value of that instance will naturally increase. The blockchain allows us to track ownership of the NFT (the single signed copy), as the sequence of transactions is immutable in the ledger. We can verify that this exact copy was first owned by the author themselves. An NFT can represent any digital item, like artworks, music, videos, in-game items, etc. and physical assets like real estate, deeds to a car, tickets to an event, legal documents, etc....

Uniqueness and guaranteed ownership are the envisioned benefits of using NFTs, but in practice, these tokens cannot hold a lot of data, so the file itself is not stored on the chain. What is actually saved is a small web link that points to a copy of the file hosted on a server. Not only this makes NFTs highly susceptible to link rot – the link becoming invalid – but also challenges the definition of NFT ownership, as server owners can simply move, delete, or change the file the link points to.

2.1.7 ERC-721

Smart contract standards can include token standards, name registries, library or package formats, and more. By defining standards, smart contracts must obey requirements in order to enable basic functions like creation of tokens, performing transactions, spending, and so on. The Ethereum Request for Comments (ERC) 721 is a data standard for Non-Fungible Tokens, meaning each token is unique and cannot be divided or directly exchanged. The ERC-721 standard allows creators to issue unique crypto assets via smart contracts. Additionally, it provides functionalities, such as transferring tokens from one account to another, getting the current token balance of an account, retrieving the owner of a specific token, or, the total supply of a specific token available on the network.

Thus, “minting” an NFT is accomplished by a smart contract on the blockchain, that takes the receiver’s wallet address and the chosen digital asset (mostly web links).

2.2 Analysis of NFT Integration in Video Games

The three main objectives that crypto enthusiasts aim to achieve by integrating Non-Fungible Tokens in video game architectures, are: true player ownership of in-game items,

interoperability of these digital assets, and, a return on investment (playing the game) by selling these assets on a marketplace. Given that game assets are, at a low level, data blocks, that can be held within NFTs, this may seem theoretically achievable. In the reality of game development, however, these mechanics are nothing less of an impossible dream.

2.2.1 Decentralized Ownership

Game items are usually formed of multiple components. Take for example a leather bag. This simple game item includes a 3d model file (fbx, obj, etc.) that describes the mesh data of the object, a mix of texture maps (color, displacement, roughness, luminosity, glossiness, etc.), material data, some animations to open and close the bag, a couple of sound files to accompany the bag's interactivity, different versions of the model (empty, full, damaged, etc.).



Figure 2.7 - 3D Leather Bag

For a complete object ownership, all these components, and more (physical attributes, collision data, etc.), that represent the single entity that is this particular leather bag, would need to be included in the token's metadata. Although technically doable, two major disadvantages render it impractical. The first being the astronomical fees associated with the transaction, execution

of the smart contract, and the cost to store data on the blockchain. Ethereum uses storage slots in the size of 256 bits. It costs 20,000 gas to store data in a slot.

There are 8 bits in a byte, thus a single slot can store 32 bytes; so, the storage cost will be around 640,000 gwei. Keep in mind that the size of the previously mentioned components is no less than a couple of hundreds of megabytes, which, given the current price of ether coins, it would cost no less than a hundred thousand U.S. dollars. And that's to mint only 1 token. The end goal of minting every special item, collected by every player, during gameplay, simply cannot be afforded.

The other disadvantage is a byproduct of the nature of the blockchain design. Games contain bugs. Whether caused by tight deadlines, undiscovered engine bugs, irreversible implementations, the bigger, and more complex games are, the more they are prone to be released with unfixed issues. That is the reality of the game industry, and a fact. And by associating game data with NFTs, which are immutable, game developers become incapable of fixing existing data. The only way to do so, would be transacting the existing tokens away from the players' accounts, then minting and sending them a new token, which was already catastrophically expensive the first time. Also note that games receive patches at least bimonthly.

The above disadvantages can be greatly reduced by simply minting a link to a server, that points at a specific game item, instead of the game item data itself. This way the developers can update the item without any overhead cost. However, this method breaks the decentralized nature of NFT, since the data is on a company owned server, and link rot can occur at any time, and the token would lose its intrinsic value.

2.2.2 Interoperability Of Game Items

Let us assume that all the previous problems were solved, and players can easily own hundreds of tokens, that can be updated at any time. The idea of interchanging items data between games, and hoping that everything would work flawlessly, is a long shot. Every game engine currently available has its own render pipeline, physics framework, custom object importer, programming language compiler, file formats (for animations, materials), graphics shaders, assets reader, and much more, which makes it impossible to import a random game asset residing in a token on the chain. In addition, the code would have to understand the structure of the data, to properly read it, and utilize it, which first has to overcome the many challenges of standardization in the game industry.

For now, let us assume that was not the case, that any game can load any game item file perfectly.

Games do not understand the physical representation of an object. Imagine a token, containing the data of a demon slaying sword, imported into a game of basketball. The game would see it as mesh data that needs to be rendered in the game space. However, the program of the game would break, as no sword swinging mechanics are found, it would not even know that the data imported represents a sword. Game developers cannot account for all types of objects, it is extremely time and budget consuming, borderline impossible.

Again, let us assume that games are smart enough to employ any type of object.

There exist no incentives for game companies to accept game items generating from other, potentially rival, games. Not only would it break the immersion of an extensively designed world (such as importing a gun to a swordsmanship game, or a colorful jacket into a black and white detective game), but it would also destroy the carefully created economy of the game.

Any modern Massively Multiplayer Online game offers a market place for the players to trade game items for game currency. The value of these items is relative to their scarcity in game, so any item that would be procured outside of this ecosystem, would cause about an inflation or deflation of the prices, which in turn affects the gameplay itself.

From another perspective, with the rise of Free-to-Play, companies became more dependent on selling in-game cosmetics. The creation of this content incurs a cost on the company. But, in return, increases player retention. For example, Epic Games, studio behind sensational hit Fortnite, which has made around 2 billion dollars, in the year 2019 in microtransactions alone, has partnered up with many studios, such as Disney's marvel, Lucas Films, etc. to bring iconic characters to its game world. Allowing other game companies to benefit from the availability of these characters, by minting them, is counterproductive. If players can buy the ironman skin from Fortnite, and use it in a game they find more fun, that game would benefit from the costs of the company to create this content, at no cost whatsoever.

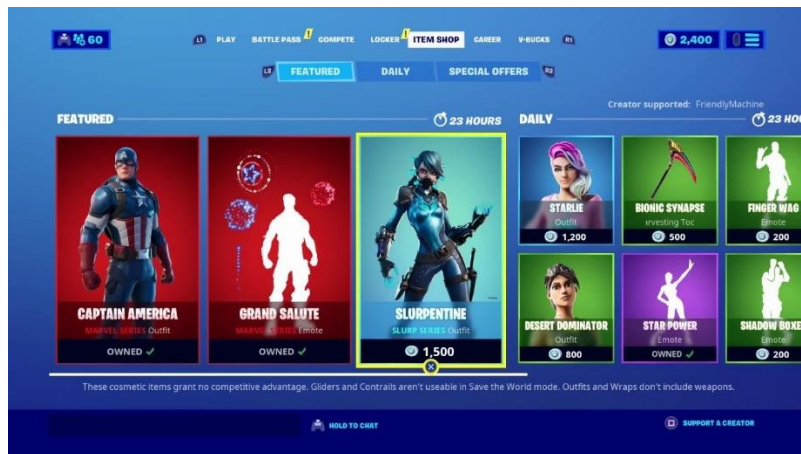


Figure 2.8 - Marvel-Based Characters in Fortnite's Item Shop

2.2.3 Play-to-Earn

Technical issues aside, integrating NFTs in video games introduces a new business model. Alongside Pay-to-Play and Free-to-Play, blockchain-based games are built upon the Play-to-Earn model. Play-to-Earn is the movement toward commercializing the act of playing video games, allowing players to make real money from their playtime and gain a “return on investment” from their game experience. Players have the ability to cash out from the game, and generate revenue, by grinding and obtaining rare items, as NFT, and selling them on the marketplace.

The economics simply don't add up. A game, such as a poker game, cannot generate wealth. By its very nature, it creates an unsustainable pyramid scheme-like system where new players buy in only to create and sell digital goods to those who buy in after them. Above and beyond that, it creates a culture of virtual indentured servitude where individuals are dragged forward by the threat of a lost investment to play a video game at the cost of their own mental and physical health. All currently active Play-to-Earn games require an entry fee from the player. Some directly, but most games require the player to pay indirectly as to acquire the needed elements to play the game.

Axie Infinity, one of the leading blockchain games, has its players purchase 3 axies to form a starting team, and battle to earn smooth love potions. An average (weak) team would cost around a hundred U.S. dollars. SLP is an item used to breed new axies, and is the main currency traded amongst players. By winning and ranking up, more SLP can be won from battles. But here is the catch, to win more battles, you need a stronger team, and to get a stronger team, players have to either breed them, using SLP, which is a gamble, since 80% of axies offspring are weaker than their parents, or purchase them from the marketplace. The

average price of a useful axie is around \$100, so forming a competing team would require at least \$300. On average, with such a team, 150 SLP can be won per day, which is equivalent to just above half a U.S. dollar.

2.3 Descriptions of the Used Software and Algorithms

The software used are discussed in more detail throughout the development process of the games. This research employed ganache to simulate a local Ethereum blockchain, solidity, a programming language to compile smart contracts, and Play Canvas, an online game engine to develop html games.

2.4 Previous Work in the Subject

As this topic is relatively novel, there aren't, to the extent of my knowledge, any paper that discusses the implications of implementing true NFT mechanics in a video game. The papers that were referenced either discuss new Consensus models, or the viability of play to earn, in games that allow simply transacting items, without employing true ownership standards, or interoperable game assets.

2.5 Research Motivation

While researching the topic, I have found many papers that present new video game-oriented consensus models, such as Proof of Play, or business models, or discuss the technical challenges of the current capabilities of available blockchains. However, I found few studies that tackled the incompatibilities between the nature and mechanics of blockchain systems and NFTs, and

the unwritten rules of game design. This thesis pinpoints several critical points to consider when developing blockchain-based games.

Chapter 3: Original Work

In the field of technology, and development in particular, there is always room for growth. Crypto and NFT enthusiasts argue that the technology is still in its early age, and just like web 2.0, it will take time to grow and become the “normal” of everyday internet. Blockchain technology by itself is very interesting, the idea of a system operating without human intervention has significant potential.

Two web games, that each present to the player multiple NFT collectibles, while providing a minimal engaging gameplay, are implemented. Functionalities were designed in a minimalistic fashion, as to solely analyze the performance and effect of an NFT system, and eliminate any collision that might occur.

3.1 NFT-Friendly Design

Not all game items should be NFTs. Items ought to have a trait of individuality. Artificial scarcity, further enhances uniqueness. The games are designed each around an item that qualifies to be minted. Crypto run, an endless runner platformer prototype, focuses its gameplay on enabling players to acquire its characters as unique tokens. All the mechanics involved are tailored tools that help players achieve that goal. The same is true for Crypto Forgers, a blacksmithing simulation game. These prototypes are in no way designed as full games, but with enough features, even if non-functional, to retain some level of realistic standards.

3.1.1 Crypto Run

Crypto Run is an endless runner platformer. The goal of the game is to gather as many apples as possible, while avoiding collision with traps. The longer a player survives, the more apples are generated and valued, and the faster the game becomes. In terms of gameplay functionality, the game is kept simple. Progression is set on a single axis, in 2 dimensions. Players have to simply focus on 1 mechanic only, jumping.

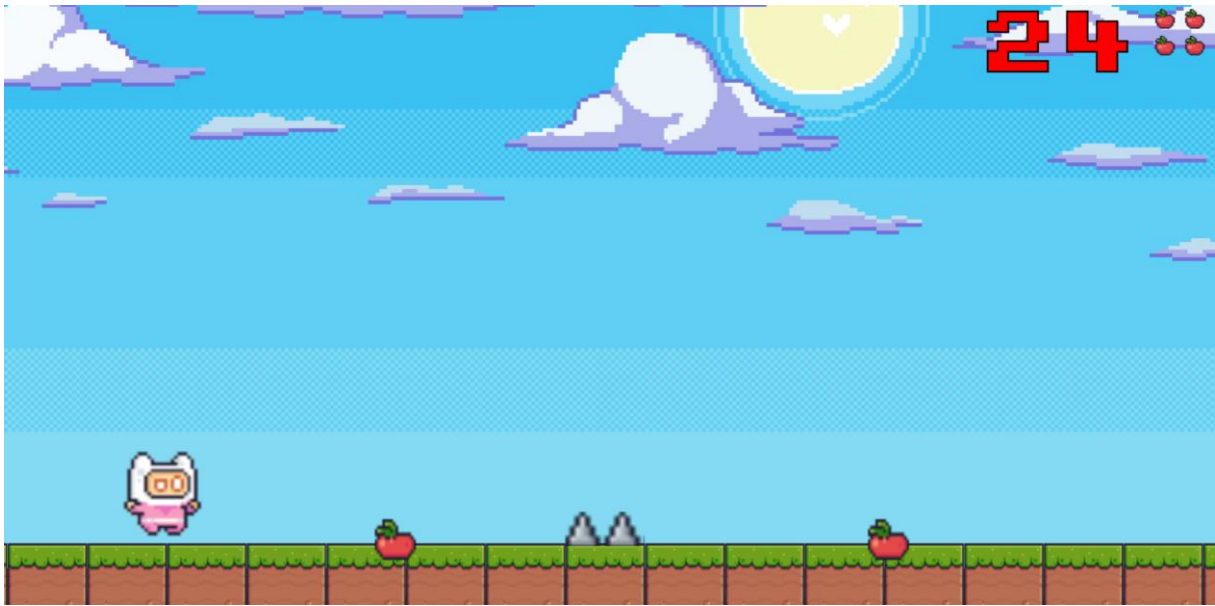


Figure 3.1 - Crypto Run Gameplay Footage

In the main menu, before starting a new game, players are prompted to select one of the characters they own, to use it in the game. Four characters were created for the purpose of this test. While theoretically better to rely on procedurally generated content, as incorporating unique, random genes into the characters, would benefit the company from a business side, creating an artificial scarcity, and thus granting items intrinsic value, the technical implications remain the same. Therefore, to keep the design simple, hand-designed characters are utilized.

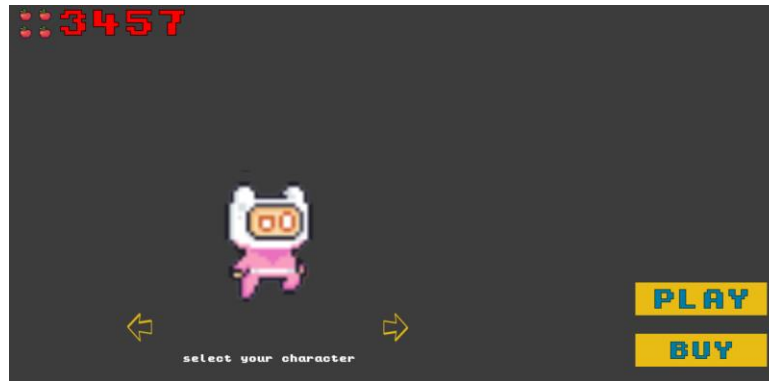


Figure 3.2 - Crypto Run Main Menu UI

At the start of the game, players do not own a character, and, as done in many modern popular play-to-earn games, have to start by purchasing a character on the store. Gathering apples allow the player to purchase a new playable character. Clicking on the buy button will select a random character to be minted as an NFT, at the cost of 100 apples. Players now truly own these tokens, as they exist on the blockchain, outside of the game's ecosystem, and they are able to trade them, or cash out on the marketplace.

3.1.2 Crypto Forgers

Crypto Forgers is a fantasy-styled, simulation idle clicker game. The player takes the role of a blacksmith, and can train and enhance their creation skills. Again, the gameplay is kept simple, to prevent complex architectures that would meddle with our study's goal. The game is presented fully as a static user interface, with 2 options for the user to click through, either forging a new weapon, or simply training. Some non-functional elements, such as a set of attributes associated with each weapon, are introduced to retain a realistic standard of content a normal game would have. The aim is not to add weapons as NFTs, but it is to incorporate NFTs into functional ecosystems. An assumption is made, that this prototype, is part of a bigger world, that utilizes weapons, which usefulness is numerically established as attributes.

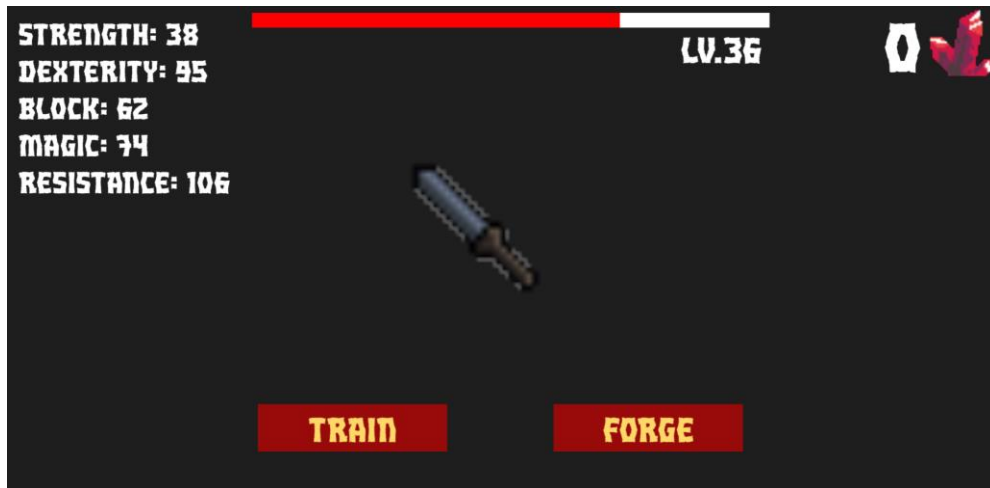


Figure 3.3 - Crypto Forgers Gameplay UI

By training, players can accumulate experience points, that allow them to increase their level. Above the level indicator, a progress bar at the top center of the interface, displays the progression made in the current level, and an estimate of the remaining ratio needed. The number of crystals held by the player, the currency employed in the game, and, some numerical representations are also displayed. Every 5 level upgrades, the player gains a crystal.

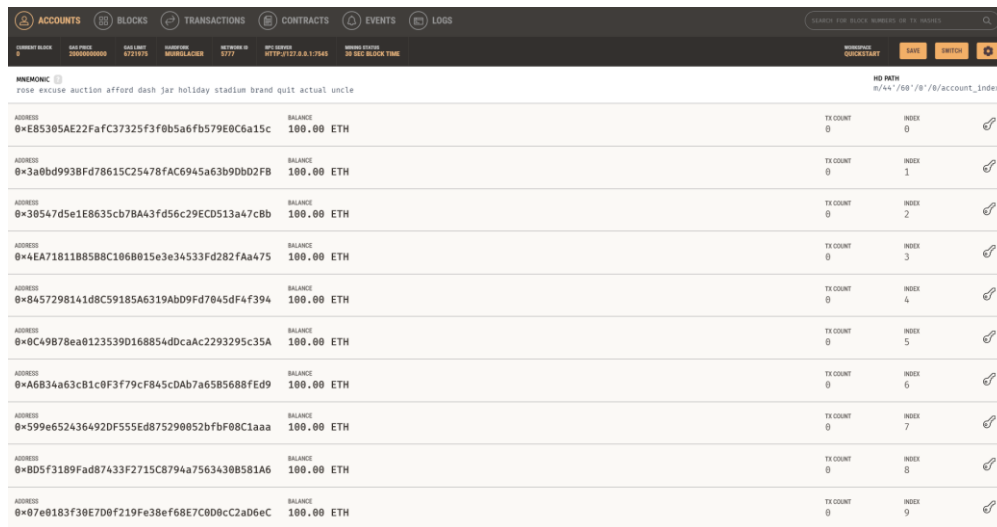
The distinct traits of the weapons: the artistic design, the real-life representation (sword, spear, shield, etc.), and the randomly weighted attributes; the combination of these 3 elements are minted into NFTs. The values of the attributes are generated with regards to the players current level, and the artwork becomes more distinguished, and rare, the higher the player's level.

3.2 NFT Integration

Gameplay implementation was quite straightforward. Now, it is important to assimilate NFT generation with the gameplay flow. Receiving tokens should not break the player's immersion.

3.2.1 Ganache

Ganache is a free software that offers a testing framework for web3 decentralized applications. It provides a local Ethereum based blockchain, with virtual wallets and miners. All functionalities of the current Ethereum chain are emulated realistically. It tracks and debugs transactions, manages deployment and execution of smart contracts, mints NFT, simulates costs and gas fees, etc.



ADDRESS	BALANCE	TX COUNT	INDEX
0xE85305AE22FaFc37325f3f0b5a6fb579E0C6a15c	100.00 ETH	0	0
0x3a0bd993Bfd78615C25478fAC6945a63b9Db02FB	100.00 ETH	0	1
0x38547d5e1E8635cb7BA43fd56c29ECD513a47cBb	100.00 ETH	0	2
0x4EA7181188588C1068015e3e34533Fd282fAa475	100.00 ETH	0	3
0x845729814d8C59185A6319AbD9Fd7645dF4F394	100.00 ETH	0	4
0x0C49B78ea0123539D168854dDcaAc2293295c35A	100.00 ETH	0	5
0xA6B34a63cB1c0F3f79cF845cDAb7a65B5688fEd9	100.00 ETH	0	6
0x599e652436492DF555Ed87529052bfF08C1aaa	100.00 ETH	0	7
0xBD5f3189Fad87433F2715C879Aa7563430B581A6	100.00 ETH	0	8
0x07e0183f30E7D0f219Fe38ef68E7C0D0cC2aD6eC	100.00 ETH	0	9

Figure 3.4 - Ganache

3.2.2 Smart Contract

To mint NFTs, in other terms, generate a token that can hold data, we need to utilize an ERC-721 contract. Solidity is an object-oriented, high-level language designed to target the Ethereum Virtual Machine, and implement smart contracts. OpenZeppelin is a library that provides the inheritable implementations of the ERC standards. The following script creates an ItemToken contract, that inherits the ERC721URISStorage base class. Every token should have its own unique Identifier. The counter class is used to simply assign incrementing Ids, the first token id will be 0, the next 1, and then 2, etc.... The constructor allows initializing the contract on the

chain. And finally, an AwardItem function is implemented. By calling this contract's function, any developer on the blockchain can mint their own data. Notice that the function takes 2 parameters, the address of the receiver of the token, and the data to be stored. As previously discussed, it is currently not feasible to save file data, such as png, jpg, fbx, etc. and only string data, like web links or JSON, can be used. After a unique Id for the token is retrieved, the function calls the inherited mint, and setTokenURI provided by the standard. The newly generated Id is returned to the caller.

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.14;
3
4 import "@openzeppelin/contracts/token/ERC721/extensions/ERC721URIStorage.sol";
5 import "@openzeppelin/contracts/utils/Counters.sol";
6
7 contract ItemToken is ERC721URIStorage {
8     using Counters for Counters.Counter;
9     Counters.Counter public _tokenIds;
10
11     constructor() ERC721("Item Token", "ITM") {}
12
13     function awardItem(address player, string memory tokenURI)
14         public
15         returns (uint256)
16     {
17         uint256 newItemId = _tokenIds.current();
18         _mint(player, newItemId);
19         _setTokenURI(newItemId, tokenURI);
20
21         _tokenIds.increment();
22         return newItemId;
23     }
24 }
```

Figure 3.5 - ERC-721 Smart Contract

To deploy the contract to the ganache blockchain, the truffle-js library is provided in the same suite of software. Using Node.js framework, we can migrate a new version of the smart contract, and deploy it. It is important to note that any data on the chain is immutable, and as such, migrating and deploying new contracts will always create a new instance, as it is impossible to update existing data.

```

1  const ItemToken = artifacts.require("ItemToken");
2
3  module.exports = function(deployer){
4    deployer.deploy(ItemToken);
5  };

```

Figure 3.6 - ItemToken Contract Migration

Of course, deploying a contract has associated fees. These fees are covered by a virtual wallet on the virtual chain. But in reality, the game company would have to cater for the costs, keeping in mind that any update to the contract will require them to cover recurring expenses.

2_deploy_contracts.js

=====

Replacing 'ItemToken'

```

> transaction hash: 0xf83b784881bb9dae521a1893a2ed8329ceeca7c0071de180b528ce6a074525bc
> Blocks: 0        Seconds: 28
> contract address: 0xb9308579abF0dCdFF7347D9df70512C20c9cf6B0
> block number:    3
> block timestamp: 1658873805
> account:         0xc7582E7cc9145BCAb2553b742E96bCeF76762e29
> balance:         99.9438424
> gas used:        2516513 (0x266621)
> gas price:       20 gwei
> value sent:      0 ETH
> total cost:      0.05033026 ETH

```

```

> Saving migration to chain.
> Saving artifacts

```

```

> Total cost:      0.05033026 ETH

```

Figure 3.7 - Contract Deployment Fees

One final thing to note is that the game code needs to call the contract's function, and therefore, needs a language it can recognize to talk to. JavaScript cannot make Solidity function calls. To circumvent this issue, the truffle library generates an abi file, that is the smart contract in JSON format. Through it, and the web3-js library, we can make function calls to mint items using vanilla JavaScript.

3.2.3 Creating and Loading Tokens

Thanks to the web3-js library, we can easily communicate with the deployed smart contract to read and create new tokens. One thing to note is that tokens created by a contract, can only be loaded through it. Simply requesting the balance of a wallet address on the chain, will not retrieve all tokens transacted to it. This creates a significant constraint on the problem at hand, if game companies wish to associate NFT assets with universal usage, then all these companies must use the same smart contract. However, we know full well that technology secrecy plays a big part in the game industry. Every company has its own software, engines, libraries, etc. and smart contracts would fall directly into this category. Especially if more than simple minting logic were to be implemented. For this research, we assume that all companies will use the same contract.

```
39 | let walletaddress = ethereum.selectedAddress;
40 | web3 = new Web3(Web3.givenProvider || "http://127.0.0.1:7545");
41 | const abi = this.contract.resource.abi;
42 | const address = '0xb9308579abF0dCdFF7347D9df70512C20c9cf6B0';
43 | const contract = new web3.eth.Contract(abi, address);
```

Figure 3.8 - Contract Connection Code Snippet

After connecting to the blockchain, hosted at port 7545, we load the contract data from the abi JSON file, define the address of the contract (present in the abi file), and we create a web3 contract object. Once established, we can now call methods defined on the contract.

To create new tokens, we can simply call the AwardItem function on the contract. We have to additionally pass the data to be saved, the receiver's address, as well as the address that will settle the gas fees of transaction and contract execution. In the two prototypes, it is up to the player to cover those costs.

```

await token.methods
  .awardItem(walletaddress, data)
  .send({ from: walletaddress })
  .on('transactionHash', (h) => console.log('hash', h))
  .then((val) => {
    console.log('value', val.events.Transfer.returnValues);
  })
  .catch(console.log);

```

Figure 3.9 - NFT Generation

Once this function is called, Metamask, a digital wallet manager, built as an extension for chrome, prompts the user to accept the transaction.

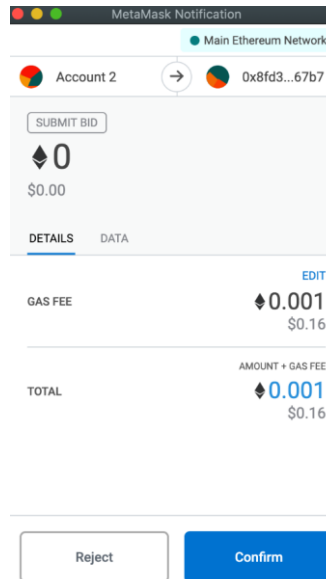


Figure 3.10 - Metamask Transaction Prompt

To load data, we can utilize the `getPastEvents` function, that keeps track of all events that occur for this contract. By specifying to simply return “Transfer” events (creating and transferring an NFT to the user), and filtering only the transactions that were sent to the current user’s wallet address, we can retrieve the list of token ids that are owned by the current user.

```
45 |     let tokenIds = [];  
46 |     await contract.getPastEvents('Transfer', {  
47 |         filter: {  
48 |             to: walletaddress  
49 |         },  
50 |         fromBlock: 0  
51 |     }).then((events) => {  
52 |         for (let event of events) {  
53 |             tokenIds.push(event.returnValues.tokenId);  
54 |         }  
55 |     });
```

Figure 3.11 - Loading Tokens Ids

After receiving the list of Token Ids, the developer must then load the data associated with each token, that is another call to be made, for every individual token.

```
57 |     for (let id of tokenIds) {  
58 |         let val = await contract.methods.tokenURI(id).call();
```

Figure 3.12 - Loading Token Data

Because every game is using the same contract, when loading player tokens, the developer will receive a complete list of every token, from every game the user has played. This can break the game on runtime. When programming functionalities, code should be aware of the structure of the data it is working with, and loading heaps of tokens, designed by unknown developers, without the capability of knowing beforehand the hierarchy of the data is disastrous. Unfortunately, the only viable solution would be to ignore data our code does not understand. However, this specificity plays against interoperability of the game assets.

Another unlikely solution is for the companies to agree on a standard, by doing so, developers would be more at ease loading and checking tokens' data. However, not only would it take competing companies to settle for common grounds, which in its way, limits creativity, and still, could not prevent malicious parties from using the contract to spam malfunctioning tokens.

3.2.3.1 In Crypto Run

The data saved in the Crypto Run NFTs is a json string, formed of two values. The first, game code, simply is an identifier I have added to recognize the game that the loaded tokens are associated with. The second value is the id of the specific character this token represents. As I cannot save full graphical and components data in the token, I can only save a reference id, that the game understands, and knows how to process. If the loaded token, for example, has an assetId value of 2, then my game understands that this is the “pink” character, and identifies the assets to load into the game, whether the image of the character displayed in the menu selection screen, or animated during the gameplay.

3.2.3.2 In Crypto Forgers

While I have also included a game code for this game’s tokens as well, the fact remains that this is an impractical method, as standardization is still an issue in the industry. Aside from the game code, the token consists also of an assetId, for the game to know what weapon visual to load, and the attributes’ values, which dictates the intrinsic value of the item. The better the attributes, the rarer the item, the more it will be sought.

Chapter 4: Conclusion

This chapter echoes the process to validate and assess the viability of an NFT model within a video game, and summarizes the different challenges highlighted by the experiment. Then, we discuss the main contributions achieved by this research. Finally, it opens up on new theories and approaches to tackle this conundrum.

4.1 Summary of the Main Results

4.1.1 The process

Two web-based video games were developed in order to assess the feasibility of real player ownership of in-game elements, interoperability of assets between multiple games, and, the validity of the play-to-earn business model. The gameplay is kept fairly simple, as to omit any complicated architecture that might affect the NFT-features' implementation. The process highlighted the natural inconsistencies between the NFT blockchain model requirements and the status quo of game development methods. Additionally, considering that all commercial games are more intricate in design, the impracticality seen in this straightforward example will inevitably be magnified by bigger games' interconnected complexities.

4.1.2 The challenges

The research exposed multiple obstacles to tackle in order to implement true player ownership of in-game items. Due to the high fees of data tokenization, and smart contract deployment, the process of minting game items proved to be tremendously costly, whether on the player or the

development studio. Additionally, the token itself does not have the capacity to hold big data. Consequently, in current practice, it simply holds a link to a hosted company owned server, that prompts the given data. This invalidates the concept of data decentralization, as the link is always at risk of “link rot”, and the data is susceptible of being modified by the company.

Aside from the technical limitations, the immutability of an NFT clashes with the normal progress of a game’s journey. After its release, a game is continuously updated with patch fixes, data optimization, and, graphics enhancements. By saving this data on a blockchain, the developers are forced to re-issue new tokens to the players, which is cost inefficient, and put the game at risk of bugs when encountering outdated data. As for the token’s value, which is based on the artificial scarcity of the data it holds, it is consistently volatile, since it is based on the value given to the item by the game. If an item becomes less rare, or useful in the game, its value will plummet. If the item is removed, or even more, the game is shut down, the tokens will completely lose their values.

This research also demonstrated the failings of interoperable game items. Assuming an NFT can hold the full extent of a game asset, loading it to another game environment is a different problem. Since game studios operate on different engines, with most of them being internal and hidden from the public, not all games read and understand data the same way. Therefore, loading data processed from a different engine is not straightforward, close to unrealistic. Trying to achieve that in the current state of games only leads to crashes, bugs and malfunctions. On the other, there are no incentives for game companies to share or accept other games’ assets. Given the fees allocated to creating new game items artistically and programmatically, then including them in the game’s ecosystem, the research fails to see the reason studios would allow other games to utilize these assets for free. Even more, since the game world is carefully drawn

up and crafted, accepting random items into it can break not only the game's economy, but also the player's immersion.

The research finally highlighted the effect of the play-to-earn model on the game itself. Players tend to optimize the fun out of a game, which prompted the designers to usually force the players in certain game directions. Adding a layer of income to the game will attract players that seek more financial gain than fun, and will affect the experience of all other players.

4.2 Main Contributions of the Thesis

This research provided a working prototype that illustrates the limitations of integrating NFT mechanics into the process of game development. It aims to show that blockchain technology is still in its early stages, and far from being a part of a multi-billion-dollar industry, and the highly optimized, semi-universal pipeline of game making.

The process showed the inability of crypto tokens to hold enough data necessary to represent in-game objects, while also being tremendously costly to mint. It also highlighted the unviable postulation about the straightforwardness of interoperable game assets. And finally, it shed some light on the ludicrous business model of play-to-earn, and its effect on the fun ratio of the game.

4.3 Possible Extensions and Future Work

While the research showed the inconsistency of integrating blockchain NFTs in games, it is not to say that blockchain technology by itself is not useful. With advances in usability, flexibility and affordability, this tech could present novel ways to create gameplay, not for the sake of

monetizing every aspect of a game, but to induce more immersive and interesting experiences to the players.

Bibliography

- Ann Aguila, D., & Miguel Bartolata, J. (2022, February). AXEing the Axie Infinity (AI): The AI of Modern Gaming, Business Model Strategem, and Global Economy Towards Cryptocurrency Era.
- Burkert, R., Horwat, P., Lutsch, R., Roth, N., Stamm, D., Stamm, F., . . . Jansen, M. (2021, September 03). Decentralized Online Multiplayer Game Based on Blockchains.
- Cai, W., Wu, F., Yin Yuen, H., C. B. Chan, H., Yan, Q., & C.M. Leung, V. (2019, July). Proof-of-Play: A Novel Consensus Model for Blockchain-based Peer-to-Peer Gaming System.
- Carvalho, A. (2021, January 26). Bringing transparency and trustworthiness to loot boxes with blockchain and smart contracts.
- Chroma Way. (2022, June). Effects of Blockchain on Game.
- Deepak Puthal, N. M. (n.d.). Everything you Wanted to Know about the Blockchain.
- Downling, M. (2021, April 29). Is non-fungible token pricing driven by cryptocurrencies?
- Erkki Siira, E. A.-K. (2017). Designing and Implementing Common Market for Cross-Game Purchases between Mobile Games.
- Gao, S., & Li, Y. (2021, October 2021). An empirical study on the adoption of blockchain-based games from users' perspectives.
- Gao, S., & Li, Y. (2021, April 27). An empirical study on the adoption of blockchain-based games from users' perspectives.

- Jiang, X.-J., & Liu, X. F. (2021, March 1). CryptoKitties Transaction Network Analysis, The Rise and Fall of the First Blockchain Game Mania.
- Komiya, K., & Nakajima, T. (2019, June 16). Increasing Motivation for Playing Blockchain Games Using Proof-of-Achievement Algorithm.
- Léo Besançon, C. F. (2019, May). Towards Blockchain Interoperability: Improving Video Games Data Exchange. IEEE.
- Min, T., Wang, H., Guo, Y., & Cai, W. (2019, August 22). Blockchain Games: A Survey. IEEE.
- Munir, S., & Baig, M. (2019). Challenges and Security Aspects of Blockchain Based Online Multiplayer Games.
- Pfeiffer, A., Kriglstein, S., & Wernbacher, T. (2020, September). Blockchain Technologies and Games: A Proper Match?
- Serada, A. (2020). Why Is CryptoKitties (Not) Gambling?
- Serada, A., Sihvonen, T., & Harviainen, J. (2020, February 20). CryptoKitties and the New Ludic Economy: How Blockchain Introduces Value, Ownership, and Scarcity in Digital Gaming.
- Solido Games. (n.d.). Solido Games – An Decentralized Ecosystem for.
- Vidal-Tomas, D. (2022, February 25). The new crypto niche: NFTs, play-to-earn, and metaverse tokens.
- Wajde Baiod, J. L. (2021). Blockchain Technology and its Applications Across Multiple domains: A Survey .

Yin Yuen, H., Wu, F., C. B. Chan, H., C. M. Leung, V., & Wei, C. (2020, October). Infinity Battle: A Glance at How Blockchain Techniques Serve in a Serverless Gaming System.