ABSTRACT

FAIRNESS IN NIL-BASED NFT MARKETPLACE (NNM)

by Monu Chaudhary

This thesis proposes a fair marketplace for exchanging NFTs representing the Name Image and likeness (NIL) of celebrity. The NIL-based NFT Marketplace (NNM) allows fans to purchase NFTs in the primary market, and exchange and auction NFTs in the secondary marketplace. The main objective of NNM is to offer a fair marketplace where celebrities are continuously rewarded with royalties on transactions of NFTs that they represent. Fairness is achieved by using two methods: randomizing the generation of NFTs and controlling the probability of generating NFTs on purchase. These methods help us achieve egalitarian fairness. While, in the case of a secondary market, since the price of transactions cannot be controlled by the system, it allows celebrities to earn rewards based on their utility function i.e. popularity, which helps us achieve envy-free fairness. We performed two sets of interviews: 1) with a group of celebrities (student-athletes), and 2) with potential end users. The first round of interview was conducted before the implementation of NNM to identify clear goals and motivations for building the solution. The second round of interview was conducted after the implementation of NNM to analyze the usability and fairness of NNM from the perspective of end-users.

FAIRNESS IN NIL-BASED NFT MARKETPLACE (NNM)

A Thesis

Submitted to the

Faculty of Miami University

in partial fulfillment of

the requirements for the degree of

Master of Science

by

Monu Chaudhary

Miami University

Oxford, Ohio

2023

Advisor: Dr. Suman Bhunia

Reader: Dr. Arthur Carvalho

Reader: Dr. Liudmila Zavolokina

Reader: Dr. Dhananjai Rao

This Thesis titled

FAIRNESS IN NIL-BASED NFT MARKETPLACE (NNM)

by

Monu Chaudhary

has been approved for publication by

The College of Engineering and Computing

and

The Department of Computer Science & Software Engineering

Dr. Suman Bhunia

Dr. Arthur Carvalho

Dr. Liudmila Zavolokina

Dr. Dhananjai Rao

Table of Contents

List of Tables

List of Figures

Acknowledgements

I would like to sincerely thank my thesis advisor, Dr. Suman Bhunia, from the Department of Computer Science and Software Engineering at Miami University. His unwavering support, guidance, and continuous feedback have been invaluable throughout this thesis. I am truly grateful for his dedication to ensuring the successful completion of my work.

I would also like to extend my gratitude to Dr. Dhananjai Rao, Dr. Arthur Carvalho, and Dr. Liudmila Zavolokina for sharing their knowledge and expertise, as well as providing valuable comments and suggestions that greatly contributed to the final outcome of my thesis.

I am deeply appreciative of my parents and friends for their constant support and encouragement throughout my academic journey. Their love and belief in me have been instrumental in keeping me motivated and determined during the entire process.

Lastly, I would like to acknowledge and express my heartfelt gratitude to all the individuals who have supported me in various ways during the completion of this thesis. Their contributions, whether direct or indirect, have played a significant role in my academic success. Thank you to everyone involved.

Chapter 1

Introduction

Since its inception, collectible cards, also known as trading cards, have been popular among enthusiasts for decades for playing collectible card games (CCGs) which involve the formation of a strategic deck and gameplay [4]. Traditionally, collectible cards are paper-based small cards that feature images of a celebrity, a famous place, or a famous object [5] usually with statistical information about the featured subjects. With digitization, paper-based cards have taken digital forms which allows interoperability and wide accessibility. Despite their popularity, they are associated with several challenges such as being counterfeited, lacking proof of authenticity and ownership, etc. In this thesis, we aim to bridge the gap and propose a non-fungible token-based digital collectibles marketplace that uses name, image, and likeness (NIL) of celebrities being featured.

In this chapter, we elaborate the open challenges faced in collectible cards marketplaces. Based on the challenges, we discuss about the solution that we propose and the novelties introduced by our solution.

1.1 Open Challenges

With the current state-of-the-art, collectible cards have taken digital forms and are available online. This has enhanced accessibility and convenience for collectors or fans and has made the experience more interactive and engaging. Although digital (non-blockchain) based collectible cards offer several advantages over their physical counterparts, they also present many challenges. Non-blockchain-based digital collectibles may suffer from issues of ownership, authenticity, and transparency.

Due to the challenges faced by the non-blockchain-based digital collectible cards, recently, there is a thrust on making collectible cards using NFT over the blockchain. Some of the advantages of blockchain-based digital collectible cards include authenticity, ownership, immutability, and transparency. This makes blockchain a suitable platform to host our smart contracts. In this paper, we investigate the feasibility of using NFTs for replacing collectible cards with a special case study of college-based athletes as celebrities. We are going to use the NIL as the main selling point of the NFTs.

We conducted a case study to understand the opinions of student-athletes on NFT marketplaces and the challenges that they are faced with to propose a fair system. From the case study, we found people divided on the concept of fairness. Some said that everyone should be treated equally while others said that people who deserve more should get more and those who deserve less should get less. More information about the case study can be found on in Section 3.1.

1.2 Proposed System

Based on the aforementioned challenges, we design NIL-based NFT Marketplace (NNM) that offers a fair chance for fans to collect and trade NFTs with other fans. On NNM, a fan can perform purchases, exchanges, and auctions of NFTs representing a celebrity. For every purchase, NNM uses our randomization algorithm which generates an NFT of a random celebrity. This guarantees a fair distribution of NFTs, each representing a celebrity, among the fans so that each fan has a variety of NFTs. Whenever a transaction is performed on NNM, the celebrities whose NFTs are traded and the deployer who is the creator of the NFTs get some rewards as royalty. Our rewarding algorithm makes sure that there is a continuous flow of royalty to the celebrities who represent the NFTs being traded and to the deployer who is the creator of NFTs. More details about the transactions and the rewarding algorithms are provided in the respective sections of this paper.

The paper then delves into the fairness of the reward earned by the celebrities. From the interviews with student-athletes (see more in Section 3.1) and potential users (see more in Section 6), we found that they are divided on whether the reward earned by the celebrities should be proportional to the popularity of the celebrities or should it be the equal regardless of the popularity. In this paper, we investigate, whether NFT based marketplace could potentially enforce correction for each celebrity to earn an equal reward from NFT exchanges regardless of their popularity. However, NFT transactions in the secondary market are based on fans who determine the price of NFT based on the popularity of the celebrity. The only thing the NFT administrator could do is control the number of NFTs in circulation for each celebrity. Controlling NFTs drawn from the primary market could then control the rewards earned by the celebrities enabling a fair distribution of the rewards. In this research, we investigate that approach and found that it is a feasible approach where we could perform correction up to 99% accuracy.

1.2.1 Novelties of Proposed System

The novelties of NNM are listed below.

1. Design a new digital marketplace for fan collectibles that allows NFT exchanges and auctions.
2. Design a live auction marketplace where fans can put their NFTs for auction
3. Royalty distribution to celebrities and the deployer whenever an NFT transaction takes place.
4. Offers fairness through the randomized generation of NFTs for purchase.
5. Designed a fairness correction scheme that can enforce egalitarian fairness (all celebrities earn an equal reward) by correcting card selling probabilities.

As per our span of knowledge, NNM is the only model which guarantees to offer fairness in the ability to earn rewards from each transaction of an NFT.

1.3 Organization of the Thesis

The rest of the paper is organized as follows. Section 2 provides a detailed background and state-of-the-art solutions. Section 3 provides information regarding the motivation for developing NNM and its goals in detail. Section 4 provides detailed information about the design of NNM. Section 4.4 gives a detailed description of the transactions that can be performed on NNM. Section 5 describes the implementation of NNM and different experiments performed to evaluate the feasibility of NNM. Section 6 describes the responses that we received from the potential users of NNM. Section 7 provides details about the fairness of NNM and the evaluation of fairness using Jain's index. Finally, in Section 8, we point out some of the future works and conclusions that we are able to draw from the research.

Chapter 2

Background & Related Work

In this chapter, we discuss the background information needed for our work. We explore the state-of-the-art in this field and relevant topics related to the work we propose in this thesis. We begin with Section 2.1 where we describe name-image-likeness. In Section 2.3, we describe blockchian network and its features. Next, we describe Non-Fungible Token and Fairness in Section 2.4.1 and Section 2.7 respectively. Finally, we briefly describe some of the related works in Section 2.8.

2.1 Name Image Likeness (NIL)

Name, image, and likeness (NIL) are the three elements of a person's right to publicity. Right to publicity requires that anybody using the name, image, or likeness of a person have permission to use it [6]. These elements including their nickname, symbols, signature, social media account, etc. collectively or independently identify a person in a unique manner [7]. These components can be referred to as a brand that a person possesses and can be used to gain financial benefits.

Common examples of the use of NIL include the sale of an celebrity's jersey, appearance in a commercial or advertisement, etc. [8].

2.2 Collectibles Card Game and Digitization

Traditionally, a collectible card or trading card is a paper-based small card that represents a celebrity, a famous place, or a famous object [5]. It usually contains a name, an image, and a short description. Fans usually collect cards for playing CCG which involves strategic deck building and gameplay where the cards are valued proportionately with their rarity [4, 9]. Fans can connect with other fans, collect their favorite cards, earn points, and compete with other fans through the internet [5]. The challenge of having a physical card or a digital card (not blockchain-based) is that the uniqueness of the card is not guaranteed and the ownership of cards cannot be proven.

The advancement in blockchain technology has enabled the use of non-fungible tokens (NFTs) as collectible cards. Non-fungibility property of a token makes it unique and avoids counterfeiting and fraudulence, ensures authenticity [9], and provides proof of ownership [10] of the token. Additionally, the fundamental properties of blockchain networks such as immutability, transparency, availability, and decentralization make them the most suitable platform to host NNM.

One of the most popular blockchain-based collectible marketplaces is NBA Top Shot which allows fans to buy, sell, and trade officially licensed NBA collectibles, such as trading cards and

highlight clips. These digital collectibles, called “moments,” are unique and can range in rarity and value.

2.3 Blockchain Network

Blockchain is an immutable, distributed, and decentralized append-only database. In a blockchain network, the database is distributed across multiple computational devices called nodes spread over different geographic locations. The redundancy of databases created over multiple nodes eliminates the problem of having a single point of failure and increases the reliability of the network. The data in the blockchain network is not controlled by a single entity which removes ownership of the data by a single entity/organization. The data in the network is stored after performing some cryptography algorithms which increase anonymity and security in the network. The append-only nature of blockchain technology makes it immutable. Any block added to the blockchain cannot be changed or deleted.

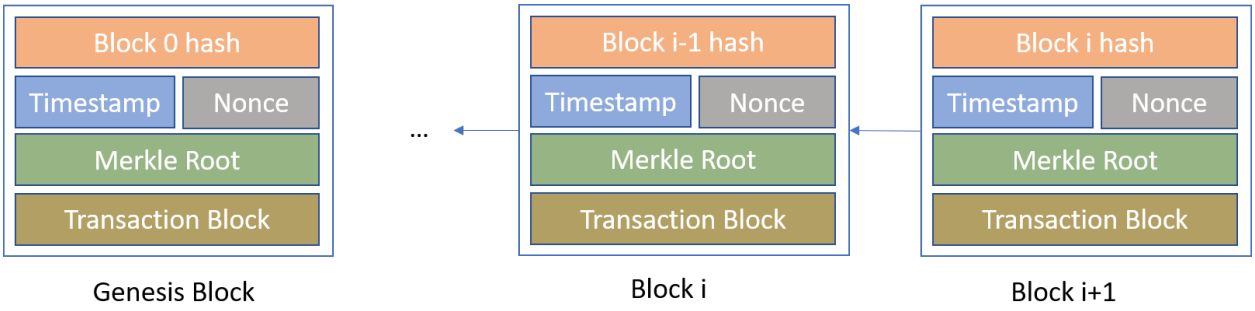


Figure 2.1: Blockchain example [3]

2.3.1 Properties of Blockchain Network

The properties of blockchain technology is described below.

- The transactions in the blockchain are validated using a **distributed consensus** mechanism which removes the reliance on a third party.
- The participation of a large number of nodes in consensus makes the transactions practically **immutable and irreversible**.
- The use of public cryptography to sign every transaction stored in the blockchain ensures **data provenance**.
- The data is distributed in a decentralized manner over multiple nodes which makes the blockchain technology **reliable**.
- Blockchain technology ensures **accountability and transparency** of the data.

2.3.2 Blockchain vs Central Database

The above features exhibited by blockchain technology make it a better choice than a central database system. The difference between blockchain technology and central database is shown in the *table 2.1*.

	Blockchain	**Central Database**
Trust	Trust ensured through consensus. No need of third party.	Needs a trusted third party.
Data Confidentiality	Data is transparent and accessible to all.	Data access is restricted.
Fault tolerance	Redundant distribution of data makes it falut tolerant.	Data stored in a central database which is not fault tolerant.
Performance	Reaching consensus is slow.	Immediate execution.
Redundancy	Each participating node has a latest copy.	No redundancy is maintained.
Security	Uses cryptography methods.	Uses traditional access control.

Table 2.1: Blockchain vs Central Database [1]

2.4 Ethereum Blockchain Network

Ethereum is an open-source, globally decentralized computing infrastructure powered by blockchain technology [11] [12]. The project was co-founded by Vitalik Buterin in 2013. Ethereum runs programs called smart contracts which are executed automatically whenever a condition is met. It uses blockchain to synchronize and record the system's state. Ether, the cryptocurrency used in the Ethereum blockchain is used to meter and constrain execution resource costs. The Ethereum network allows developers to build secured decentralized applications with built-in economic function [11].

Some of the advantages of ethereum blockchain network are listed below.

- **Data coordination:** In an Ethereum blockchain network, information, and trust are better allocated across network participants removing the need for a trusted central authority to manage the system [13].
- **Data Immutability:** The core feature of blockchain makes data in the Ethereum network immutable securing it from data corruption.
- **Removes central authority:** The capability of Ethereum to execute smart contracts on EVM makes it an autonomous entity removing the need for a trusted intermediary.
- **Fast transaction:** The automation of the transactions and validity of the transactions makes it faster than the lengthy manual process.

- **Secure transactions:** All transactions in the Ethereum network are secured cryptographically.
- **Reliablity:** Ethereum offers high reliability by eliminating the risk of a single point of failure. Use of the Ethereum network removes drawbacks of centrally hosted applications such as downtime, censorship, fraud, third-party interference, etc. [14].
- **Turing completeness:** It implies that the Ethereum network can compute anything computable given enough resources.
- **Stateful transactions:** Unlike bitcoin, which only deals with transactions, the Ethereum network has the capability to store states of different parties involved, balances, etc [14].

2.4.1 Non Fungible Token (NFT)

Unlike fungible tokens, non-fungible tokens are tokens that are unique in existence and cannot be interchanged with another token. Non-fungible tokens have gained popularity since the advent of different blockchain technologies. Some of the examples of NFTs are collectibles, tickets for an event, digital art, an original version of a song/music, tokenized real estate, etc. No two NFTs are the same in features. Even multiple copies of the same non-fungible token will be uniquely identified from others and can be obtained as a single collectible [15]. Whereas fungible tokens such as Bitcoin can be exchanged for another Bitcoin without any loss of value. Linda Xie, the co-founder of Scalar Capital and former Coinbase product manager defines NFT as a *"unique digital asset whose ownership is tracked on a blockchain"* [16].

It is of paramount importance to correctly authenticate an asset. In the environment where cards of dubious authenticity are spread all around, non-fungible tokens (NFTs) have provided a significant and reliable solution to forgery. The properties of NFTs ensure that each NFT is uniquely identifiable through cryptography code [15] and can be reliably associated with the owner without any doubt.

The unique characteristics of the NFTs help to track the ownership of a digital or a physical asset. The ownership of an NFT is recorded in the blockchain and can be transferred by the owner, allowing NFTs to be sold and traded. Blockchain networks such as Ethereum provide a track of ownership by issuing digital certificates of ownership associated with the NFT when an NFT is encoded or minted onto a blockchain network [17]. This uniqueness of NFT when coupled with blockchain enables the ability to control, transfer and issue ongoing royalties or other payment streams to the creator of the NFTs.

2.4.2 Smart Contract

Since the advent of Bitcoin [18] by Nakamoto in 2008, different concepts or models of blockchain have come into popularity. One such model is smart contract. A smart contract is a contract between two or more parties programmed electronically and executed automatically [19] over the underlying blockchain platform [20] whenever an event in the contract is triggered from the user end. The

distributed and tamper-resistant nature of blockchain has made it a perfect platform for hosting smart contracts. Ethereum is the most popular platform to host smart contracts.

Smart contracts offer the following benefits.

- Automatic execution of the algorithms ensures the autonomy of management processes and thus removes the need for a trusted third party. For example, an automatic transaction of assets, transfer of digital rights between multiple parties, etc.
- Immutable data log that offers information regarding all the transactions.
- Faster speed of processes that need manual efforts to run.
- Smart contracts use encryption to keep data secure on the blockchain level.
- Automated execution of smart contracts eliminate the intervention of human on the process level. This automation makes the whole transaction less prone to error.
- Smart contracts are deployed and executed on decentralized and distributed blockchain architecture which reduces the risk of single-point failure.

These advantages of smart contracts have increased the popularity of the contracts in recent years. As of February 2020, over 1 million smart contracts were deployed in Ethereum. This trend has resulted in about 2.7K decentralized applications (DApps) offering services in the sector of health, finance, games, gambling, etc. [20].

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.4.22 <0.9.0;

contract Migrations {
  address public owner = msg.sender;
  uint public last_completed_migration;

  modifier restricted() {
    require(
      msg.sender == owner,
      "This function is restricted to the contract's owner"
    );
    _;
  }

  function setCompleted(uint completed) public restricted {
    last_completed_migration = completed;
  }
}
```

Listing 2.1: Example of Smart Contract

Life cycle of Smart Contract

The life cycle of a smart contract consists of four phases. The four phases are discussed below [21].

1. **Creation of smart contracts:** The rules, obligations, rights, and prohibitions written in natural language is converted into smart contract written in computer languages. This phase includes design, implementation, and validation sub-phases.

2. **Deployment of smart contracts:** The next phase includes the deployment of the smart contract on top of a blockchain network. Once a smart contract is deployed, it cannot be modified. A new contract should be deployed to implement any changes.

3. **Execution of smart contracts:** After the deployment of the smart contract, the users can trigger the contractual event whenever a contractual condition is met. A breach of contract leads to automatic punishment of the respective party.

4. **Completion of smart contracts:** After the execution of a smart contract, the state of the involving parties is updated and the committed transactions are added to the blockchain.

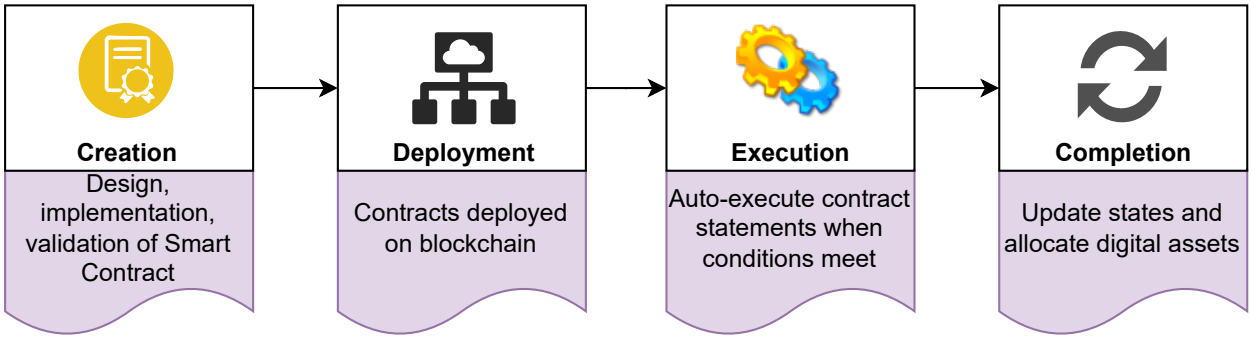


Figure 2.2: Smart Contract Life Cycle

2.4.3 Solidity

Solidity is a high-level contract-oriented language with similarities to JavaScript and C language[22]. Smart contracts deployed on the Ethereum blockchain are written in solidity. The smart contracts are compiled to EVM bytecode for deployment. Solidity is currently the flagship language of Ethereum. *listing 2.1* shows an example of a smart contract written in solidity language.

2.4.4 EVM

Ethereum Virtual Machine (EVM) is an Ethereum execution environment for building and managing smart contracts [23] [21]. EVM is a decentralized network of computers that processes millions of programs - smart contracts. Smart contracts, written in solidity high-level programming language are converted to bytecodes after compilation via a compiler (*e.g.*, solc for solidity). The bytecodes are then run on top of the EVM [21].

2.4.5 Gas

Gas fees are the cost used by the Ethereum network to execute a transaction. In the Ethereum blockchain, the gas fee is paid in its native currency, ether (ETH). The Ethereum gas price indicates

the cost of performing a computation. It also helps developers understand energy consumption against smart contract code [23].

2.4.6 DApp

A DApp (decentralized application) is an application developed with a smart contract on the back-end, in lieu of a conventional database and web application hosting provider [22]. In practice, DApps act as universally available web services running on the EVM. Users can, however, access the DApp via HTML/CSS/JavaScript front end accessible through a web browser or a smartphone application, or an Ethereum browser [22].

2.5 Consensus

A consensus algorithm is a fault-tolerant mechanism that ensures that an agreement is reached among all the peers of a blockchain network on the state of the distributed ledger. The consensus mechanism enables trust and reliability in the blockchain network. The objectives of a consensus mechanism are to reach an agreement, collaboration, cooperation, equal rights for every node, and mandatory participation of each node in the consensus mechanism.

2.5.1 Objectives of Consensus Mechanism

- **Unified agreement:** Unlike a centralized system, a decentralized system lacks a trusted entity that is responsible to ensure the integrity of the system. A consensus mechanism is an algorithm that ensures trust in a trust-less blockchain system [24].
- **Align economic incentives:** The trustless system of blockchain requires all the interests of miners in the network to validate each transaction. The incentivization policy of the consensus mechanism offers rewards for good behavior and punishes the bad actors [24].
- **Fair and equitable:** The blockchain consensus mechanism enables all participants to equally participate in the network.
- **Prevent double-spending:** In a blockchain, only the transactions that have been verified and validated are stored. This prevents the double spending problem i.e. the currency is spent twice.
- **Fault tolerant:** The blockchain network is consistent, fault-tolerant, and reliable, that is, the blockchain network would work indefinitely without any downtime.

2.5.2 Proof of Work (PoW)

Proof of work consensus mechanism is used in Bitcoin and Ethereum (before the merge in 2022 [25]). The nodes participating in the PoW consensus mechanism are called miners and the PoW

procedure is called mining [26]. PoW consensus mechanism selects a miner who would validate the generation of the next block in the chain. The miners in the process compete to solve complex mathematical puzzles using comprehensive computation power. The one who solves the problem at the earliest is rewarded with the next block [24].

2.5.3 Proof of Stake (PoS)

Unlike PoW, where participants are required to invest their time and energy as well as buy some expensive mining equipment to solve complex mathematical puzzles. PoS requires validators to lock some of their coins in the system as stakes to buy proportionate block creation chances [27] [24]. The PoS consensus mechanism deterministically chooses the creator of a new block based on the proportion of stake. A participant with a larger stake has a higher probability of validating the creation of a new block. For example, a participant with a stake of 10% has a 10% probability of validating the next block. For each block validated, the validators are paid transaction fees. For this reason, participants who own a larger stake receive a greater reward than the participants with a small stake [27]. In PoS, no new coins are minted or mined when transactions are validated.

2.6 Crypto Wallet & Metamask

Cryptocurrency software wallets, often known as crypto wallets, are used by crypto holders to securely store their digital currencies or cryptocurrencies and tokens in one place. Users can make use of cryptocurrency wallets to perform various transactions which include buying, swapping, lending, and earning cryptocurrency.

MetaMask is a software cryptocurrency wallet that offers an interface to interact with the Ethereum blockchain [28]. Metamask wallet is available to users in two forms: (1) browser extension, and (2) mobile app; either of which can be used to interact with decentralized applications running on Ethereum virtual machine. Metamask offers essential features to its users: (1) store and manage account keys, (2) broadcast transactions, (3) send and receive Ethereum-based cryptocurrencies and tokens, and (4) securely connect to DApps.

2.7 Fairness

2.7.1 Algorithmic Fairness

The term fairness is difficult to define as the concept of fairness varies depending on the actual contexts. Fairness is a social concept. Thus, technologies targeting algorithmic fairness should be approached from a socio-technical perspective. The socio-technical perspective acknowledges the influence of both technical and social structures on the system's outcome.

Algorithmic fairness is a wide area of discussion in the context of systems making decisions either inferred from data or expert knowledge [29]. In the paper [30], Verma and Rubin introduced the concept of fairness-aware programming. Fairness in decision-making programs should be designed so that the program shows no bias towards certain groups of users.

2.7.2 Fairness in Smart Contract

The fairness of a smart contract can be defined only when the aspects that make a smart contract unfair are identified. A smart contract is unfair to a group of users if the expectation of the users does not match with the actual implementation of the game rules [20]. Fairness issues are introduced due to the logical design of the smart contracts. One of the examples is the introduction of Ponzi schemes in games that do not offer the rewards that it falsely states to offer. Even many DApps which claim to have a fair auction algorithm may still have a possibility of bidders colluding among themselves or with the auctioneer to make a profit at the expense of others [31]. The fairness issues are introduced through contract logic because of the unfair design or due to some careless mistakes. Since fairness is a subjective matter, a contract fair to someone may be unfair to others.

The fairness of a smart contract can be quantified based on the four fairness properties: truthfulness, efficiency, optimality, and collusion-free.

- *Truthfulness* exists in smart contracts when the stakeholders have an equal probability of benefiting from a transaction. Given an auction smart contract where the auctioneer, the one who creates the auction, puts an NFT on auction. Many bidders can join to compete with each other for the NFT. If the auction prevents the bidder from benefiting more by bidding less, it is truthful.

- *Efficiency* is achieved when no bidder can influence the bid number of other bidders. The winner is the one who truly values the bid and sets the highest bid.

- *Optimality* of a smart contract depends on the result of the bid. If the one who bid the highest value is always the winner, then the auction is said to be optimal. Optimal auction enables the auctioneer to maximize its profit.

- *Collusion-free* auction means that no bidder can collude with any other bidder to achieve a higher profit. A collusion-free smart contract helps to prevent manipulation of the bid price to an extent. This helps to ensure that there is a fair chance for all bidders to win and the auctioneer gains good revenue.

2.7.3 Threat to Contract Fairness

One of the ways smart contracts can become unfair to the participants in an auction mechanism is through shill bidding. In shill bidding, bidders collude with each other or the auctioneer to escalate the price of the item without any intention to buy it. This will result in a high bidding price which other participants may think of as fair and may have to pay a higher price for the bid.

In a survey by Bartoletti et al. [32], the authors found that about 0.05% of Ethereum transactions could belong to Ponzi schemes. Ponzi schemes violate the fairness properties since such contracts do not offer an equal opportunity of gaining profits. Che et al. [33] conducted research on the identification of patterns of Ponzi schemes in the smart contract of a decentralized application. In order to identify Ponzi schemes in smart contracts, [33] used a machine learning and data mining approach to train a classifier.

2.7.4 Unfair Smart Contract

The unfairness of a smart contract can be a result of multiple factors: (1) the Absence of logic, (2) Incorrect logic, and (3) Logically correct but unfair [34].

Absence of logic

```
function bid(uint32 auctionID) external payable {
        AuctionInfo memory info = auctionIdToInfo[auctionID%MAXVAL_32];

        auctionIdToInfo[auctionID%MAXVAL_32].highestBidder = payable(msg.sender);
        auctionIdToInfo[auctionID%MAXVAL_32].highestBidAmount = msg.value;

    }
```

Listing 2.2: Example of Bid contract with missing logic.

The above code snippet is for a bid contract where the value of the highest bidder and the amount bid by the highest bidder is being updated. But, the code snippet lacks the check for whether the amount bid by the bidder is the highest or not. Due to this, any bidder can bid a smaller amount than the last bid.

Incorrect logic

```
function bid(uint32 auctionID) external payable {
        AuctionInfo memory info = auctionIdToInfo[auctionID%MAXVAL_32];

        require(info.isOpen == true);
        require (block.timestamp < info.bidExpiryTime);
        require (msg.value > info.highestBidAmount);

        if (info.highestBidder != address(0)) {
            info.highestBidder.transfer(info.highestBidAmount);
        }

        auctionIdToInfo[auctionID%MAXVAL_32].highestBidAmount = msg.value;

    }
```

Listing 2.3: Example of Smart Contract

The above contract is logically incorrect because it doesn't update the address of the bidder. When the bid is complete, the bid amount is not transferred to the auctioneer.

One of the popular contracts, HackersGold, had a bug in the transferFrom function. The function had a typographical error where the addition assignment operator (+=) was written as =+. Due to the error, the recipient did not receive the incremented balance. In the MultiSig smart contract, the function to initialize the owner of the wallet was inadvertently made public. Exploiters took advantage of the error by changing the owner of the wallet causing a huge loss of money.

Logically correct but unfair

```
function bid(uint32 auctionID) external payable {
        AuctionInfo memory info = auctionIdToInfo[auctionID%MAXVAL_32];

        require(info.isOpen == true);
        require (block.timestamp < info.bidExpiryTime);
        require (msg.value > info.highestBidAmount);

        if (info.highestBidder != address(0)) {
            info.highestBidder.transfer(info.highestBidAmount);
        }

        auctionIdToInfo[auctionID%MAXVAL_32].highestBidder = payable(msg.sender);
        auctionIdToInfo[auctionID%MAXVAL_32].highestBidAmount = msg.value;

    }
```

Listing 2.4: Example of Smart Contract

In the United States, by law, an auction can be *"with reserve"* or *"without reserve"*. In an auction *"with reserve"*, the seller or the auctioneer is allowed to place a bid. The participation of the seller in the bidding process can affect the willingness of the participants. The seller can influence the bid amount by placing a high bid value. Unsuspecting bidders with no expertise in analyzing code may lose their money due to artificially increased bids or forfeit their participation fee.

2.7.5 Fairness in NFT Ecosystem

The fairness of NFT is dependent on the fairness of the smart contract. Fairness is subjective and depends on the preferences of the participants. A smart contract considered fair by one party may be considered unfair by another party. To take into account such nuances, each party/stakeholder should be modeled before defining the properties of fairness of the model [20].

2.7.6 Analysis of Fairness in NFT Ecosystem

Exploitable fairness

A large portion of decentralized applications in the blockchain space are centered around gambling games where participants wager with non-fungible tokens (NFTs). These games, similar to gacha games, operate under the assumption that the cards or NFTs drawn are randomly generated, ensuring that each participant has an equal chance of obtaining a rare and valuable item. However, many decentralized applications have fallen short in delivering true randomness when minting NFTs. Consequently, individuals with advanced technical knowledge have exploited this situation, allowing them to acquire the rarest NFTs from a collection [35]. This creates an unfair disadvantage for honest participants, depriving them of the opportunity to achieve an equitable profit.

Gas auctions

The increase in gas price due to a large number of participants minting at the same time highly influences the gas price. Some of the NFT launches in 2021 disrupted the network as the launch

employed a fixed-price, first-come-first-served (FCFS) policy. The excessive demand and low price led to high competition to acquire the NFTs. One of the examples is The Seven NFT drop with an initial price of 0.07 ETH per NFT. The high number of participants escalated the gas price with median participants paying approximately 1.49 ETH and the top 5% paying $2.44+$ ETH [35].

High Skill

With high technical skills, technically adept users often try to scrape out rare NFTs leaving the honest participants with no clue of the activity. These participants directly interact with the smart contracts often bypassing frontends and mempool [36].

Exclusive Mintig

The value of an NFT collection is usually assessed by evaluating the concentration of tokens among holders. Low concentration collections are the most ideal over the high concentration collections where most of the tokens are owned by whales. NFT launches allow batch minting, where participants can mint multiple tokens at once in a single transaction and incentivizes whales with less gas overhead. This reduces the fairness in distribution as it favors those with larger wallets, especially when it comes to how expensive minting can be [35].

2.7.7 Attributes of Fair Marketplace

Things that must be considered to build a fair smart contract are explained in this section below.

Token generation algorithm

NFTs generated at each point of time should be random. No individual should be able to predetermine the features of the token. Each token in an NFT token ecosystem carries certain value based on the features of the token. When an individual is able to predict features of a token being generated, the individual can manipulate the way it wants the token to be generated and may receive an advantage over others. The ability to predict the new token doesn't make the token generation random at all. People can leverage this to mint tokens which are of high values and gain monetary benefit.

Readability of Solidity Code

Solidity is a language which is used to develop a smart contract. A smart contract is an automated contract which automatically executes itself when a condition of a contract is fulfilled. In the space of blockchain, the solidity code for most of the gaming algorithms is published publicly. This enables people to view and analyze the algorithm. People who are highly familiar with programming, especially in Solidity will have an advantage over others who find difficulty understanding the solidity code. This difference in readability or comprehension leads to unfairness when an individual with proper understanding of the algorithm competes against someone without the information [37].

Auction Mechanism

An auction mechanism should allow everyone an equal opportunity to profit from the auction without any external influences such as collusion affecting the auction process. Collusion-free auction ensures that no bidder can collude with any other bidder or the auctioneer to achieve a higher profit. Collusion-free smart contract helps to prevent manipulation of bid price to an extent. This helps to ensure that there is a fair chance for all bidders to win and the auctioneer gains a good profit. Also, there should be enough time when an item is put for auction to the time bidding starts to allow enough time for people to get informed and decide on bidding or not.

Availability/Scarcity of token

Scarcity of token determines the value of a token. A prior study is required to be performed before determining the scarcity of tokens. Fair judgement on the features of the token to be made scarce is essential.

Choice of blockchain network

The factors like gas fee, transaction time affects the participants' ability to participate in a transaction. An evaluation of the pros and cons of a potential blockchain network is essential to identify the network most suitable for deploying the decentralized application. This helps to reduce the cost of a transaction and helps the participants benefit the most.

2.8 Related Work

This section presents the related work on NFT-based marketplaces.

2.8.1 NFT-based Marketplaces

CryptoKitties is a digital collectible gaming platform based on the non-fungibility characteristics of its token. Each token i.e. cat is unique from another cat [38]. Players in the platform can buy, sell, trade and breed digital cats that the user actually owns [39]. The attributes of each cat are called *"cattributes"* which determines the amount of price it can be traded with. A high-level *"cattribute"* yields an expensive cat.

2.8.2 NFTs based on NIL

The NIL of celebrities have been highly popularized in the NFT marketplace. An NFT associated with a vintage baseball card of Hall-of-Fame shortstop Honus Wagner was sold for $3.25 million in October 2020. The value of the card was due to it being rare, unique, and the talent of Wagner as a player.

In late 2020, the National Basketball Association, in partnership with Dapper Labs - a blockchain company - launched a digital marketplace of collectible NFTs. The marketplace has been reported

to generate over $500 million in sales with more than 800,000 fan accounts [40]. The marketplace allows fans to collect virtual cards consisting of NFTs on the Top Shot website consisting of video highlights of NBA players officially licensed by the NBA. The NFTs purchased by the fans can be showcased or resold by the fans. The cards have certified authenticity and scarcity [40].

2.8.3 NFT Auction Platforms

There are multiple NFT auction platforms that allow fans to buy and sell NFTs through an auction-based system. Opensea [41] is one of the most popular blockchain-based digital collectibles marketplaces. It employs two auction mechanisms:

1. *Sell to the highest bidder* also known as English Auctions are the most common auction type on the platform where buyers bid against each other for an item, and the highest bidder wins at the end.

2. *Sell with a declining price* or Dutch auction which starts with a high price that keeps declining gradually until someone purchases the item.

Rarible is another famous blockchain-based digital collectible marketplace where users can create, buy, and sell NFTs [42]. It also allows two auction mechanisms:

1. *Timed Auction* allows the seller to set the starting and ending time, the minimum price, and sell the NFT to the highest bidder at the end of the auction.

2. *Open Auction* allows anyone to bid for the auction while the seller makes the decision whether to accept or decline the bid.

However, these auction-based platforms do not offer a mechanism to reward celebrities with royalties on transactions performed in the secondary market.

2.8.4 Fairness in Digital Space

Dolata et al [43] discuss algorithmic unfairness and the implications that it could have. An unfair system could negatively impact different stakeholders involved in the operation. The stakeholders may include (1) individuals or groups of individuals who are subject to discrimination, (2) societies at risk of social stratification and political riots, and (3) companies or organizations at the risk of getting bad publicity and/ or legal consequences. A fair decision by an algorithm should consider these social perspectives. Thus, the authors define algorithmic fairness from a socio-technical perspective.

Fairness in Smart Contract

In the research paper by Sako et al [37], the authors mainly discuss the aspects of smart contracts that make them unfair. The authors argue that it is important to identify what the unfair aspects of smart contracts are to specify what a fair smart contract will be. The authors performed a case

study on CryptoKitties, a game application built upon a blockchain network using the Solidity programming language. The authors identified that factors such as Proof-of-Stake, secret trading, blockchain anonymity, and the fee of transactions affect the fairness of a smart contract algorithm.

According to [44], fairness is obtained when each stakeholder — clients, facilitators, publishers — are fulfilled their expectations based on the application protocol. Also, timeliness plays an integral role to obtain a fair smart contract.

Chapter 3

NNM Goals and Motivation

We adopted a methodology known as *design science* [45], which involves a series of iterative steps. These steps include: 1) Identifying problem and motivation, 2) Defining objectives of the solution, 3) Designing and implementing the solution, 4) Demonstrating its functionality, and 5) Evaluating its effectiveness. In this section, we perform steps 1 and 2 of the design science approach to identify the problem that would then shape solution. Here, we first perform a case study of a specific group of celebrities to identify problems and motivation. Next, we list out the deficiencies of current-state-of-art solutions and establish the objectives for developing NNM.

3.1 Problem Identification using Student-Athletes Case Study

In this section, we delve into the case study we conducted with student-athletes, who represent a specific group of celebrities. We opted to focus on student-athletes for our case study due to their accessibility within the scope and time limitations of our research.

The case study has been reviewed and approved by the Miami Research Ethics and Integrity Office, under protocol ID 04242e. Following the approval, undergraduate, and graduate students who are also athletes at the same time opted in as participants in response to email invitations. The eligibility criteria for users to participate in the study are as follows:-

- The participant should be a minimum of 18 years of age.
- The participant should be a student student and an athlete at the same time.
- The participant should be enrolled in a university within the United States.

We conducted 12 semi-structured interviews with student-athletes from four different academic institutions in the United States. The interviewees were selected based on the criteria of being both students and athletes, with a focus on ensuring diversity in terms of gender and sport type. The main objective of the interviews was to gather the perspectives of student-athletes regarding the new NIL regulations and NIL-based NFT marketplaces. The interviews were conducted over Zoom[1] and were recorded, transcribed, and analyzed using MAXQDA software[2].

The main purpose of the interview was to identify if all the student-athletes are able to equally benefit from the NIL related endorsements and to identify areas to improve in case of NIL endorsements through NFT-based marketplace. To conduct the interview, we prepared some questionnaire

[1] https://zoom.us/

[2] www.maxqda.com

Table 3.1: Information about the interviewees (student-athletes)

Interviewee	Sport	Gender
S1	Golf	Male
S2	Golf	Male
S3	Track and Field	Male
S4	Track and Field	Male
S5	Soccer	Female
S6	Soccer	Female
S7	Track and Field	Female
S8	Soccer	Female
S9	Track and Field	Female
S10	Track and Field	Male
S11	Track and Field, Cross Country	Female
S12	Swimming	Female

to help us structure the interview process. Table 3.2 shows the questionnaires where questions $1-3$ are related to student-athletes and NIL regulations for student-athletes, and questions $4-7$ are related to NFTs and NIL.

From the interviews with student-athletes, we found that most of the student-athletes are highly positive about the NIL regulatory changes which allows them to earn money through endorsements. However, some of the them also believe that the athletes who belong to the most popular sports or athletes who are popular at other platforms such as social media will be the ones who are going to be benefited the most and that will be very few in number.

For example, interviewee S10 said the following:

> "*I guess the sports that make the most revenue normally. So, I feel like athletes, who are in bigger name sports — such as football or basketball — that have like a large following and attention from the media* [are more likely to get NIL deals]."

After the first 3 sets of questions, we proposed the concept of NNM and then asked rest of the questions that are related to NIL and NFT. The purpose of asking these questions is to potentially understand and identify solutions to NIL related problems such as lack of fairness and transparency. When we asked Question 4, interestingly, all interviewees suggested that the proposed solution has the potential to allocate financial resources to all student-athletes. For example, interviewee S2 said:

> "*It* [the solution] *has the potential to benefit everybody, not just, you know, the star athletes. So, I like it*."

Further, moving to Question 5 on potential barrier to adoption of NNM, we found it to be the novelty of the underlying technology, as suggested by interviewee S5:

Table 3.2: Interview Questions for Student-Athletes

Q. No.	Questions
1	What is your opinion on the recently proposed changes to name-image-likeness (NIL) regulations by the NCAA?
2	Among all the student-athletes, who do you think will benefit the most financially from the NCAA's recent changes to NIL regulations?
3	How can the Miami RedHawks, or any other intercollegiate athletic team, leverage the recent changes to NIL regulations to maximize the benefits received by student-athletes?
4	Do you think NNM is useful to reward all student-athletes? Why (not)?
5	What are the potential barriers to adoption of NNM?
6	Do you think NNM will allow for more fairness when it comes to exploring the recent changes in NIL regulations? Why (not)?
7	What are other potentially positive or negative aspects of the proposed solution?

> "*I don't think a lot of people know about cryptocurrency and NFTs and name-image-likeness. So, I think just advertising it* [the solution] *in a way where it is easy for all to comprehend would be super effective*."

An additional issue that was brought up pertained to the establishment and long-term maintenance of a community, specifically regarding the motivating factors for buyers (fans) when acquiring collectibles, as interviewee S11 questioned:

> "*I mean* [the solution] *obviously works perfectly for athletes whose faces are on the cards, but a quick question: what do people who purchase the cards get from that*?"

Regarding Question 6, the feedback on the fairness aspect of our solution was predominantly favorable. As an illustration, interviewee S6 provided a positive response when considering whether popular student-athletes would face any drawbacks due to the random nature of our solution:

> "*If, you know, one of these famous athletes' cards went out. Say the original pack was bought for a lower price than what the actual value of the card was worth because it was just in this random pack. But then, once it gets traded* [in a secondary market] *I'm sure there will be plenty of money to make up the difference.*"

This feedback strengthened our belief in the hypothesis that the direct sales or exchange of collectibles, which provide royalties in secondary markets, leads to a just distribution of financial resources.

Finally, the responses we received from the interview helped us in two key ways: 1) to identify the challenges associated with the adoption of NIL-based NFT marketplaces, and 2) to formulate our research goals.

3.2 Deficiencies in Conventional Collectible Marketplace

Based on the responses that we received from the interview with student-athletes as well as based on the facts that we collected from various resources, we identified some deficiencies with current NFT-based collectible marketplaces. Some of the deficiencies that we found are described below.

1. **No Royalty from Secondary Transactions** The conventional marketplaces do not offer a solid mechanism for the celebrities representing the NFT to be benefited from the transaction of the NFT on the secondary marketplace.

2. **Lack of Fairness** Fairness is a concept of *"fair or impartial treatment"* [46] which means that all are treated equally. Conventional marketplaces do not offer an equal opportunity for all celebrities to have their NFTs purchased. Most of the interviewees acknowledged that famous celebrities are the ones who are benefited the most from NFT transactions, as most buyers are interested in their NFTs. An interviewee from the case study said:

   > "*I think, naturally, some cards will have more value than others, just because of people's. You know their status or whatever within the athletic world or whatnot.*"

3. **Lack of Transparency** Conventional collectible marketplaces do not offer a trace of the transactions that they execute eluding users from viewing the details of the transactions. Also, the transactions in traditional marketplaces cannot be traced back to the genesis or origin of the NFT. Lack of transparency leads to concerns relating to the ways the systems operate and distribute royalty.

4. **Lack of Trust** The two transacting fans in a conventional collectibles marketplace cannot trust each other with the ownership of the NFTs. They depend on a central authority or a trusted third party to facilitate and execute transactions.

5. **Reliance on Central Authority** In order to establish trust between the fans on a conventional system, the fans rely on a third party. It is not guaranteed that the third party will always be fair. Also, reliance on central authority possess a risk of a central point of failure.

3.3 Objectives of NNM

After we understand the deficiencies in the conventional collectible marketplace, we outline the goals and break them into two categories: requirements and features as described below.

3.3.1 Requirements

Requirements are the essentials for NNM that overcomes the deficiencies of the conventional collectible marketplace. We have identified some of the fundamental requirements of NNM which are described below.

1. **Royalty Distribution** The interviews conducted with the student-athletes suggest that it is important to distribute royalties to both the celebrities who represent the NFTs and the marketplace owner/deployer on every transaction of NFT. This ensures a fair distribution of profits and incentivizes both parties to continue participating in the ecosystem.

2. **Reliability** Marketplaces which are built upon centralized platforms are highly prone to a single point of failure. It is essential to build a system that does not rely on a central point of failure and is robust enough so that it can perpetually exist without intervention.

3. **Trust and Transparency** The NFT owners are obligated by a central authority to verify their identity in order to access their NFTs. This restricts the owners from independently validating their ownership of the NFTs and subjects them to the policies of the central authority, which may change without considering the owners' interests. Therefore, a trustworthy and transparent decentralized system should be used which runs autonomously without dependence on the central authority.

3.3.2 Features

Features are what we want to achieve that would enhance NNM functionalities.

1. **Fairness** Fairness ensures that all celebrities who represent NFTs are treated equally, without any bias or preferential treatment. This helps to build trust and confidence among fans and celebrities and increases continuous participation.

   Fairness is a feature of NNM now, however, it can become a requirement in the future.

Chapter 4

NNM Architecture

NNM offers a platform that enables celebrities to be continuously rewarded with royalties on each transaction of NFTs representing the celebrity. It aims to offer a fair platform to all celebrities by enabling randomness of NFT generation on purchase and fairness in the distribution of rewards among the stakeholders. The use of NFTs enables NNM to achieve traceability, authenticity, and ownership of each NFT. NNM offers a scalable model to perform multiple transactions simultaneously without creating bottlenecks.

This chapter provides an overview of the NNM architecture, entities, and properties.

4.1 NNM Overview

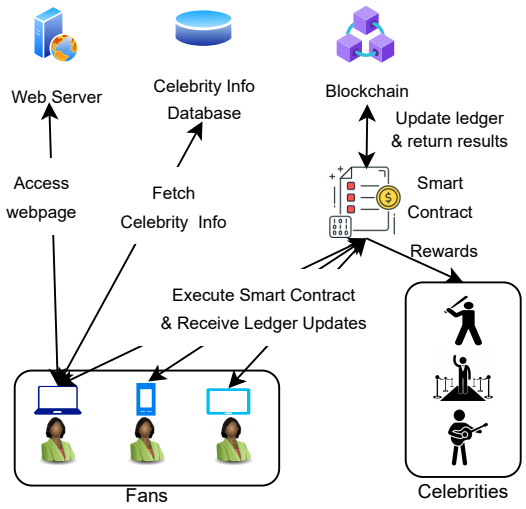


Figure 4.1: NNM System Diagram

Figure 4.1 shows an overview of the NNM architecture. NNM is a blockchain-based decentralized application (DApp) where each NFT represents a celebrity. The targeted end users are represented as fans who are connected to NNM through their cryptowallets. Each fan executes a smart contract deployed on the Ethereum blockchain network by interacting with NNM. The blockchain network stores the state of the smart contracts. Each fan accesses the information on the blockchain network through NNM web pages. The celebrity Info database is an off-chain database that stores

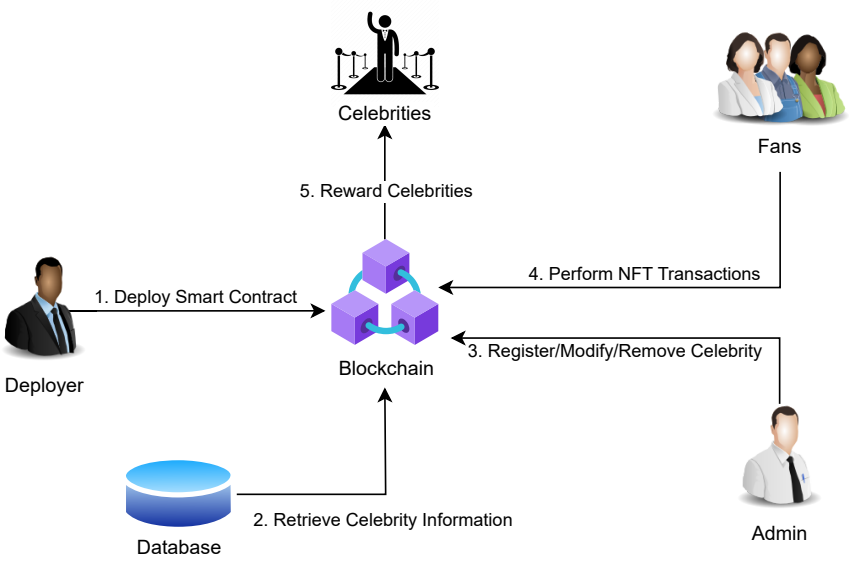


Figure 4.2: NNM Entities Interaction Diagram

name, image, and relevant information of each celebrity. On every transaction performed on NNM, celebrities represented on the transacted NFT are rewarded with some royalty.

4.2 Participating Entities

In NNM, we have four entities which interact with the blockchain network using their unique address and private key to perform NFT transactions.

1. **Celebrities** are entities that represent one or more NFTs on NNM. The total number of NFTs that a celebrity can represent is proportional to the popularity of the celebrity. When an NFT is purchased or sold, the celebrity representing the NFT is rewarded with a percentage (ϵ_c) of the total transaction amount. The amount is represented as celebrity royalty in this paper.

2. **Fans** are the potential users who can buy an NFT of a random celebrity in the primary market and sell an NFT probably making a profit in the secondary market. Fans sell their NFTs by exchanging them with other fans or by performing auctions.

3. **Deployer** or marketplace owner is the entity that deploys the main smart contract of NNM on the blockchain network. To deploy a smart contract is to create the smart contract and register the smart contract with the blockchain network. On NNM, the deployer is incentivized with deployer royalty which is a percentage (ϵ_d) of the total transaction amount for creating NFTs and providing a platform to perform transactions.

4. **Admin** is an entity that has the authority to register celebrities to the NNM, modify the celebrity information and remove the celebrity from the NNM. The deployer may assign additional admins.

5. **Blockchain escrow** is an entity that owns all smart contracts of NNM. Whenever a transaction is in progress, the blockchain escrow holds the NFT(s) and/or the amount (in Gwei) as an escrow. Once the transaction is validated on the blockchain network, the NFT(s) and the amount are transferred to the designated fans.

Figure 4.2 shows the relationship between different entities on NNM. At first, the smart contract is deployed on a blockchain network by NNM. Next, Admin is allowed to register new celebrities to NNM, modify or remove existing celebrities from NNM. The registered celebrities' information is stored in an off-chain database. In the next step, the fans can perform purchase, exchange or auction NFTs. Upon every successful completion of a transaction, the celebrities associated to the NFTs being transacted are rewarded.

4.3 Properties of NNM

This section provides the detailed properties of NNM based on the goals outlined in Section 3.3.

1. **Provide royalty on each transaction:** The transactions in NNM are performed on NFTs representing a celebrity. Each NFT is programmed in such a way that royalties are distributed to both the celebrity that represents the NFT and the marketplace owner i.e. the deployer on every transaction of the NFT. Section 4.4 provides a detailed description of each transactions that can be performed on NNM and how royalties are distributed upon each transaction.

2. **Transparency:** All the transactions that are performed on a blockchain network are visible to all the participants, ensuring transparency and accountability.

3. **Reliability:** Data on the blockchain network is distributed over multiple nodes in a decentralized manner which eliminates the risk of a single point of failure. The use of the blockchain network by NNM removes drawbacks of centrally hosted applications such as downtime, censorship, fraud, third-party interference, etc. [14] and offers high reliability.

4. **Reward fairness:** At NNM, our objective is to achieve fairness through the implementation of randomization. When a fan acquires an NFT on our platform, they are gifted with a randomly generated selection, guaranteeing equitable treatment for all. This means that each fan has a fair opportunity to possess a diverse collection of NFTs featuring various celebrities. Equally important, each celebrity is given a fair chance to have their NFTs purchased, promoting a balanced and just system. Section 7 provides a detailed description of the implementation of fairness on NNM and its evaluation.

5. **Sense of ownership:** Since NNM is based on a blockchain network in which every transaction is digitally signed by the owner using their private key. This private key can be used by the owner to provide proof of ownership [47]. While the public key which is derived from the private key is publicly available and can be used to verify the authenticity of transactions and digital assets.

4.4 NNM Transactions and Royalties

NNM ecosystem allows fans to purchase NFTs in the primary market, and exchange and auction in the secondary market.

1. *Purchase* feature allows the fans to buy an NFT representing a random celebrity.
2. *Exchange* feature allows two fans to exchange their NFTs with each other or exchange an NFT with money or both.
3. *Auction* feature allows the fans to put their NFTs for auction.

In the rest of the section below, we describe these transaction in detail.

Table 4.1: Symbols and their terminologies used in this paper

Terminology	Symbol
Fan	f
Settler (One who pays)	f_s
Initiator (One who receives money)	f_i
Auctioneer	f_a
Highest bidder	$f_{highestBidder}$
Celebrity	c
Celebrity owned by Settler	c_s
Celebrity owned by Initiator	c_i
Celebrity owned by Auctioneer	c_a
Deployer	d
Purchase Price	$\texttt{PP}$
Exchange Price	$\texttt{PE}$
Bid price	b
Minimum Bid Price	b^{min}
Instant Exchange price	b^{max}
Fraction of Reward	ϵ
Fraction of Reward earned by c	ϵ_c
Fraction of Reward earned by deployer	ϵ_d
Profit	π
Profit earned by c	π^c
Profit earned by c for a purchase	$\pi^c_{p_j}$
Profit earned by c for an exchange	$\pi^c_{e_j}$
Profit earned by c from an auction	$\pi^c_{a_j}$
Profit earned by c for all purchases	π^c_p
Profit earned by c for all exchanges	π^c_e
Profit earned by c from all auctions	π^c_a
NFT	$\texttt{NFT}$
NFT owned by Settler	$\texttt{NFT}_s$
NFT owned by Initiator	$\texttt{NFT}_i$
NFT owned by Auctioneer	$\texttt{NFT}_a$
Transaction ID	γ
Exchange Duration	t_e
Auction Duration	t_a
Popularity of c	θ_c
Correction Interval	$\texttt{CI}$
Probability Correction Exponent	$\texttt{PCE}$
Reward Popularity Exponent	$\texttt{RPE}$

4.4.1 Randomize NFT Purchase

NNM offers a platform to perform purchases on its primary market. In the primary market, the NFTs are generated and purchased for the first time. Each NFT represents a celebrity which is randomly determined by our probability distribution function during purchase. The cost of purchase (PP) is fixed for all the NFTs. Algorithm 1 shows the algorithm for performing purchase.

When a purchase is made by a fan, a new NFT representing a random celebrity is minted on the blockchain network. The NFT is assigned to the wallet address of the fan who is performing the purchase. The celebrity who is featured in the NFT is automatically rewarded with $\texttt{PP}\epsilon_c$ amount as a celebrity reward. $\texttt{PP}\epsilon_d$ amount is rewarded to the deployer (d) as a deployer reward for offering the platform to perform the transaction where $\epsilon_c + \epsilon_d = 1$.

Algorithm 1: Algorithm for NNM NFT Purchase Mechanism

1 Fan f performs a purchase with a purchase price PP
2 A random celebrity c is selected by the smart contract in accordance with the probability distribution function
3 NFT representing c is minted on the blockchain network
4 Add NFT to the list of NFTs owned by f
5 Decrease the probability of selecting c for purchase
6 Transfer ϵ_cPP to c
7 Transfer ϵ_dPP to d

4.4.2 NFT Exchange Between Two Fans

In the secondary market, NNM allows the fans to perform three different types of exchanges. Two fans can choose to exchange their NFTs with or without some digital currency. On the other hand, one fan can choose to sell their NFT to the other fan for some digital currency. Either way, it involves the transfer of ownership of an NFT from one fan to another. Algorithm 2 shows the algorithm for performing an exchange.

The three kinds of exchanges that can be performed on NNM are described in detail in the subsequent section below.

1. **Sell an NFT**

   A fan can directly sell their NFTs to another fan using the exchange feature. Let us assume the fan (f_i) initiates the transaction to sell its NFT ($\texttt{NFT}_i$) while the fan (f_s) settles the transaction on the blockchain by paying the transaction amount (PE) for the exchange using NNM. Prior to this action on blockchain, both f_i and f_s discuss $\texttt{NFT}_i$ to be sold, negotiate the exchange price (PE), and agree on the duration (t_e) for which the exchange will be active. After both f_i and f_s reach consent, f_i initializes the exchange by invoking an API call to the smart contract. In this call, details such as f_i, f_s, $\texttt{NFT}_i$, PE, and t_e are passed to the smart contract. After the exchange is initialized successfully, $\texttt{NFT}_i$ is transferred to the smart contract to be held as an

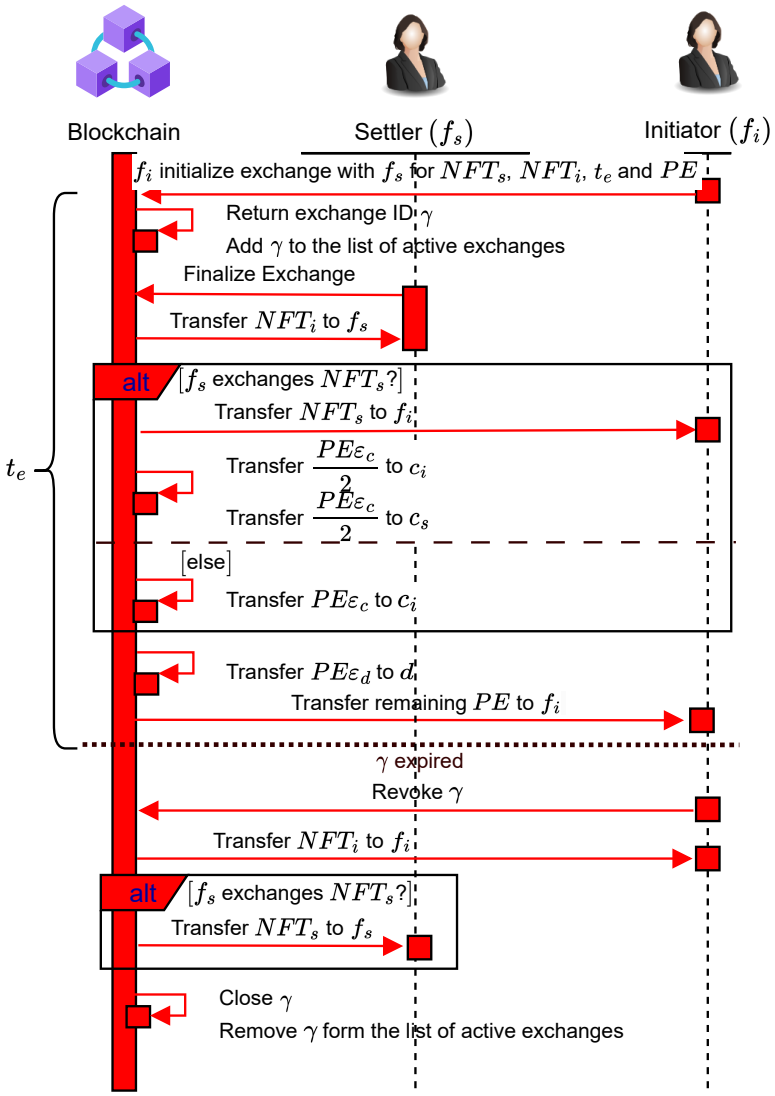


Figure 4.3: NNM NFT Exchange Sequence Diagram

escrow, and the smart contract returns a unique ID (γ) for the exchange and adds γ to the list of active exchanges.

f_s uses γ to get the details of the exchange and to verify the transaction. Once verified, f_s settles the exchange by finalizing/completing γ. Upon completion, $\texttt{NFT}_i$ is transferred to f_s, $\texttt{PE}\epsilon_c$ amount is transferred to the celebrity c who is featured in the $\texttt{NFT}_i$ as celebrity reward, $\texttt{PE}\epsilon_d$ is transferred to d, and the remaining amount is paid to f_i. Finally, γ is removed from the list of active exchanges.

In case f_s does not settle γ within t_e, f_i can recall $\texttt{NFT}_i$ from the smart contract which held it as an escrow. In that case, the smart contract transfers $\texttt{NFT}_i$ to its original owner who is f_i, and removes γ from the list of active exchanges.

Algorithm 2: Algorithm for NNM NFT Exchange Mechanism

Fans decide the NFTs and who will pay money to whom. Whoever is paying money becomes the exchange settler (f_s), and the other fan becomes the exchange initiator (f_i)
Fan f_i initializes the exchange by calling the smart contract function specifying f_s, $\texttt{NFT}_i$, $\texttt{NFT}_s$, exchange duration t_e, and/or PE which returns a unique exchange ID γ and adds γ to the list of active exchanges
if γ *is not expired* **then**
    f_s finalizes γ after verifying
    f_s gets $\texttt{NFT}_i$
    if f_s *exchanges* $\texttt{NFT}_s$ **then**
        f_i gets $\texttt{NFT}_s$
        Transfer $\texttt{PE}\epsilon_c/2$ amount to c_i representing $\texttt{NFT}_i$
        Transfer $\texttt{PE}\epsilon_c/2$ amount to c_s representing $\texttt{NFT}_s$
    else
        Transfer $\texttt{PE}\epsilon_c$ to c_i representing $\texttt{NFT}_i$
    Transfer $\texttt{PE}\epsilon_d$ to d
    Transfer the remaining amount to f_i
else
    f_i revokes γ
    Transfer $\texttt{NFT}_i$ to f_i
    if f_s *exchanges* $\texttt{NFT}_s$ **then**
        Transfer $\texttt{NFT}_s$ to f_s
Remove υ from the list of active exchanges

2. **Exchange two NFTs only**

   Unlike the exchange feature for selling an NFT, in this exchange two fans on NNM can exchange both of their NFTs in the secondary market without any exchange price.

   Before the transaction takes place, two fans (f_i and f_s) decide on the NFTs ($\texttt{NFT}_i$ and $\texttt{NFT}_s$) to be exchanged along with the duration (t_e) for which the exchange will stay active. After making the decision, f_i initializes the transaction by invoking an API call to the smart contract. In this call, the details of the exchange that include f_i, f_s, $\texttt{NFT}_i$, $\texttt{NFT}_s$, and t_e are specified. After the exchange is initialized successfully, the smart contract puts $\texttt{NFT}_i$, and $\texttt{NFT}_s$ into escrow, generates and returns a unique ID (γ) representing the exchange, and adds γ to the list of active exchanges.

   f_s performs the verification of the exchange using γ and finalizes/settles the exchange. Upon settlement, $\texttt{NFT}_i$ and $\texttt{NFT}_s$ held as escrow is transferred to f_s and f_i respectively.

   In case f_s does not settle γ by t_e, f_i can recall its celebrity $\texttt{NFT}_i$ from the smart contract. This process transfers $\texttt{NFT}_i$ to f_i, and $\texttt{NFT}_s$ to f_s, and removes γ from the list of active exchanges.

In this type of exchange, celebrities do not receive any reward on completion since there is no digital currency involved in the exchange.

3. **Exchange two NFTs with digital currency**

   Similar to the above exchange type, in this type of exchange, two fans can exchange their NFTs with each other along with some transaction amount. This type of exchange is useful when the two fans performing the exchange determine the NFTs being exchanged have different values.

   Before the transaction takes place, two fans decide on which NFTs they would exchange, the price to be paid for the exchange (PE), and the exchange duration (t_e). Once that is decided, the fan who will be paying PE becomes the exchange settler (f_s), and the other becomes the exchange initiator f_i. f_i initializes the exchange by making a call to the smart contract function specifying f_s, $\texttt{NFT}_i$ (NFT to be exchanged by f_i), $\texttt{NFT}_s$ (NFT to be exchanged by f_s), PE, and t_e. The smart contract puts $\texttt{NFT}_i$, and $\texttt{NFT}_s$ into escrow, generates and returns a unique ID γ for exchange, and adds γ to the list of active exchanges.

   After initialization of the exchange and before it expires, f_s can verify and finalize/settle γ. Upon settlement, the smart contract transfers $\texttt{NFT}_i$ to f_s, and $\texttt{NFT}_s$ to f_i. The celebrities representing $\texttt{NFT}_i$ and $\texttt{NFT}_s$ receive $\texttt{PE}\epsilon_c/2$ each as celebrity reward, the deployer receives $\texttt{PE}\epsilon_d$ amount as deployer reward, and the remaining amount is paid to f_i. The smart contract then removes γ from the list of active exchanges.

   If the exchange γ is not settled within t_e, f_i can recall its celebrity from the smart contract escrow. Upon performing the recall, the smart contract transfers $\texttt{NFT}_i$ to f_i, and $\texttt{NFT}_s$ to f_s, and removes γ from the list of active exchanges.

   For j_{th} exchange of an NFT representing celebrity c using this type of exchange with an exchange price $\texttt{PE}_j$ is represented as $\pi^c_{e_j}$ and is shown by equation 4.1.

$$\pi^c_{e_j} = \texttt{PE}_j\epsilon_c/2 \tag{4.1}$$

4.4.3 Auction-based Live NFT Marketplace

To perform an exchange, we need a separate channel to communicate. In the auction mechanism, fans can put their NFTs on auction, and other fans can directly view them and bid on them or directly purchase them without communicating beforehand.

In the auction mechanism, an auctioneer (f_a) can place an NFT ($\texttt{NFT}_a$) for auction by specifying the minimum bid price (b^{min}) and the duration of the auction (t_a). They also set a price for instant purchases (b^{max}) where anyone willing to buy the NFT can instantly purchase it instead of waiting for the auction. This price is usually much higher than the b^{min}.

Algorithm 3 shows the algorithm to perform an auction. Auctions are done on an escrow system with an auction ID. f_a initializes an auction by calling the smart contract specifying $\texttt{NFT}_a$, b^{min},

b^{max}, and t_a. Upon initialization, $\texttt{NFT}_a$ is transferred to the smart contract to hold as an escrow. Then, the smart contract generates a unique auction ID (γ) and adds γ to the list of active auctions.

Any bidder who wants to bid can only bid if the bid price $b > b^{min}$. Every time a bid is accepted, b^{min} gets updated to the new bid amount b. This allows the fans to perform only bids with a bid price greater than the previous bid amount. Also, b is transferred from the bidder to the smart contract to hold as an escrow, and the previous bid amount is transferred back to its original owner.

There are three possibilities an auction could end up with.

1. If b is more than b^{max}, the NFT is sold to the bidder.
2. If the bidding time is expired, then the NFT is sold to the bidder with the highest bid.
3. If there is no bid for the NFT by the expiry time, the NFT is revoked by f_a.

After the bidding period ends and the auction has the highest bidder, $\texttt{NFT}_a$ is transferred from the smart contract escrow to the winning bidder $f_{highestBidder}$. The celebrity representing $\texttt{NFT}_a$ receives $b^{min}\epsilon_c$ amount as a reward, the deployer d receives $b^{min}\epsilon_d$, and the remaining amount is paid to f_a. Finally, γ is removed from the list of active auctions.

If there is no bid within t_a, f_a can revoke the auction. The smart contract then transfers $\texttt{NFT}_a$ to f_a, closes the auction, and removes γ from the list of active auctions. The reward earned by c upon the success of an auction a_j with the winning bid price ${b_j}^{min}$ is:

$$\pi^c_{a_j} = {b_j}^{min}\epsilon_c \tag{4.2}$$

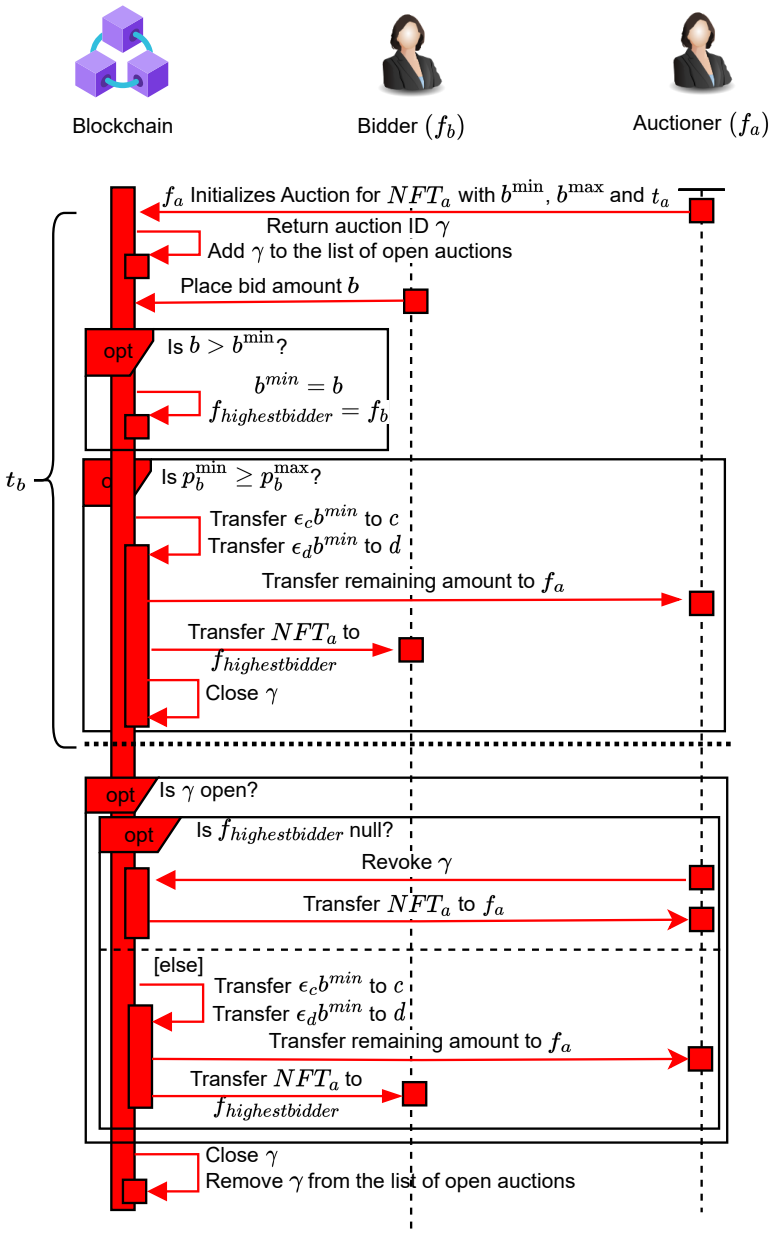


Figure 4.4: NNM NFT Auction Sequence Diagram

Algorithm 3: Algorithm for NNM Auction Mechanism

Auctioneer f_a initializes by calling smart contract function specifying $\texttt{NFT}_a$ represented by a celebrity c, bid duration t_a, minimum bid price b^{min}, instant purchase price b^{max} which return a unique auction ID γ and adds the auction to the list of open auctions
Bidder f_b places a bid with amount b
if γ *is open and* γ *is not expired* **then**
    if $b > b^{min}$ **then**
        $b^{min} \leftarrow b$
        $f_{highestBidder} \leftarrow f_b$
        if $b^{min} \geq b^{max}$ **then**
            Transfer $\texttt{NFT}_a$ to $f_{highestBidder}$
            Reward $\epsilon_c b^{min}$ to c.
            Reward $\epsilon_d b^{min}$ to d.
            The remaining amount is transferred to f_a
            Remove γ from the list of open auctions
    else
        Discard b
if γ *is open and is expired* **then**
    if $f_{highestBidder}$ *is null* **then**
        Revoke γ by f_a
        Transfer $\texttt{NFT}_a$ to f_a
    else
        Transfer $\texttt{NFT}_a$ to $f_{highestBidder}$
        Reward $\epsilon_c b^{min}$ to c
        Reward $\epsilon_d b^{min}$ to d
        The remaining amount is transferred to f_a
Remove γ from the list of open auctions

Chapter 5

System Implementation and Evaluation

Based on the design described in Section 4, we implement NNM as a distributed Application (DApp). Figure 5.1 shows the implementation architecture. The figure shows client in the front who makes connection to the backend node server performing RESTful calls. The NIL database stores the information of the celebrities which are fetched by the browser using HTTP Get request. The metamask wallet installed int the browser is used to perform smart contract transactions on the blockchain network. We are also hosting the application online for public access to fans at https://miamicards.sec.csi.miamioh.edu/. The source code implementation is published on GitHub[1]. A detailed implementation description is provided in the subsequent section.

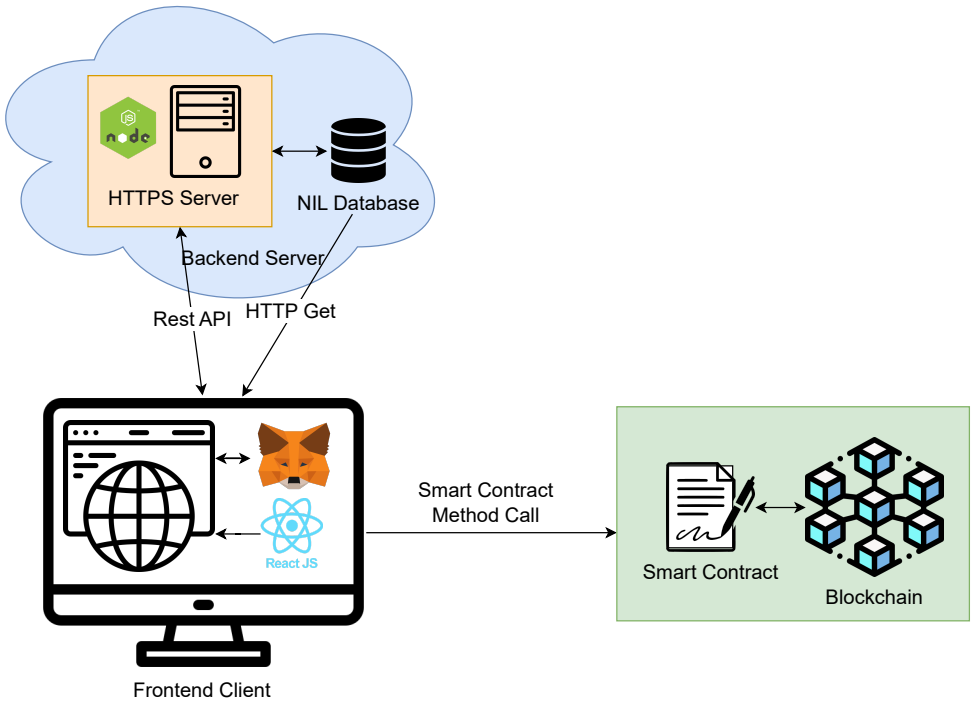


Figure 5.1: NNM System Architecture

[1]https://github.com/wsl-miami/miamicards

5.1 Prototype Development Tools

1. **NFT and Smart contract**

   There are different standards used to develop digital tokens. ERC-20 standard defines an open standard to create fungible tokens while ERC-721 and ERC-1155 standards from OpenZeppelin [48] define the open standard to build non-fungible tokens on blockchain network [39]. ERC-1155 provides APIs that are gas-efficient. Hence, we use the ERC-1155 [39] for creating NFTs and performing transactions on them.

   We used solidity programming language to build our smart contracts. Solidity is one of the most widely used programming language to build smart contracts. To implement NNM, we created 3 smart contracts, each serving its own purpose. The detailed implementation of the smart contracts are discussed in Section 5.2.

2. **Deployment on Ethereum and Avalanche Test Network**

   To discover the most optimal and cost-efficient blockchain network for maximizing the performance of NNM, we deploy it on various test networks that employ different consensus mechanisms for transaction validation and block creation. We utilize Ethereum test network. Ethereum is one of the leading blockchain platform known for its smart contract capabilities and extensive ecosystem of decentralized applications (dapps) and development tools [49]. Ethereum, which relied on PoW consensus mechanism before 2022, now uses PoS consensus mechanism which makes it even more favourable for deploying smart contract.

   We also deploy NNM on Avalanche's Fuzi C-chain test network that relies on PoS consensus mechanism. Avalanche provides fast, secure, and scalable platform for dapps while remaining highly accessible and cost-effective [50]. In Section 5.5.1, we evaluate the gas price on each of these platforms and compare them to find appropriate blockchain network.

3. **Authentication using Crypto Wallet**

   Our implementation uses a browser extension of MetaMask, a crypto wallet, that allows users to store their NFTs, cryptocurrencies, and private keys securely in one place and interact with NNM[28]. Metamask wallet provides each user with a unique wallet address which is used to authenticate NFTs' owners and to uniquely identify fans to perform transactions.

4. **Name Image Database**

   As per our design, we require a database to store celebrity information including their name, image, and description. To ensure public accessibility and optimize processing time and storage costs, we have opted for an off-chain database. Currently, in our prototype development phase, we are storing celebrity information in a collection of JSON files within the project directory.

5. **Front-end Technology**

   We use the JavaScript React library to build an interactive and dynamic user interface [51]. We use Reactstrap [52] - a bootstrap-based React UI library that offers reusable components

to build seamless and interactive web pages. We leverage the useDApp hooks [53] to perform asynchronous calls to the smart contract methods and consumed the response using async/await.

6. **Secure Web Hosting**

   NNM is hosted on a web server and we use HTTPS (Hypertext Transfer Protocol Secure) for securing the API requests to the smart contract. We use the Nginx web server which provides high performance and stability [54]. We secure the HTTP requests by employing SSL/TLS certificates [55].

5.2 Smart Contract Implementation

In this section, we describe the smart contracts that we created to implement the full functionalities of NNM. Table 5.1 shows all the functions specific to each smart contract and the details regarding the purpose of each function. The detailed description of each smart contracts is discussed in the subsequent section.

5.2.1 NNM Smart Contract

The NNM Smart Contract is main smart contract upon which other smart contract depends. In this smart contract, we define details about the deployer of the smart contract, the structure of celebrity object and how each celebrity's NFT is associated to its owner. This smart contract defines methods to register, modify, and remove celebrities from NNM. We also define some utility methods that helps us get additional details about each celebrity and each NFT. Figure 5.2 shows the methods, attributes, structs, and events used to create the smart contract. Appendex A.1 shows the solidity code implementation of the smart contract.

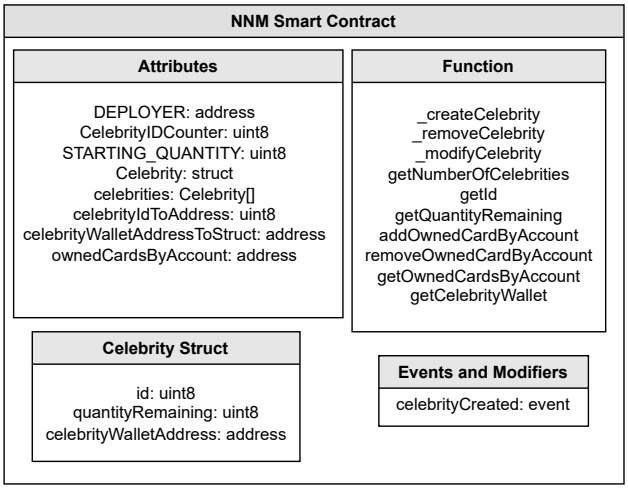


Figure 5.2: Smart Contract for registering celebrities

Table 5.1: Smart Contract methods implemented on NNM

Smart Contracts	Callers	Functions	Purpose
NNM	Admin	_createCelebrity	Function for registering new celebrities to the system
	Admin	_modifyCelebrity	Modify the details of the registered celebrity in the system
	Admin	_removeCelebrity	Remove the registered celebrity from the system
	Internal	getNumberOfCelebrities	Get the number of celebrities registered
	Internal	getId	Get the ID of the celebrity
	Admin	getQuantityRemaining	Get the remaining number of NFTs available for purchase for a celebrity
	Internal	addOwnedCardByAccount	Adds an NFT to the account of the buyer
	Internal	removeOwnedCardByAccount	Removes an NFT from the account of the seller
	Internal	getOwnedCardsByAccount	Returns the NFTs earned by a celebrity
	Internal	getCelebrityWallet	Returns the wallet address of a celebrity
CelebrityNFT	Buyer	buyNFT	Mints a new NFT of a random celebrity directly to the buyer
	Internal	randArrayIndex	Returns a random index pointing to a celebrity in a list of celebrities
	Initiator	initializeExchange	Initializes an exchange of at most two celebrities and the value difference amount with a buyer; returns an exchangeID
	Settler	finalizeExchange	Finalize the exchange of at most two celebrities and transfer the celebrities and the exchange price to the intended fans
	Settler	getFriendPostedCelebrity	Get the celebrity ID that the Initiator has posted for exchange
	Settler	getFriendAddress	Method to get the Initiator address from the exchange ID
	Settler	getCelebrityYouGive	Method to get the celebrity ID that Settler exchnages from the exchange ID
	Settler	getPriceYouPay	Method to get the price paid by Settler from the exchange ID
	Settler	getExchangeInfo	Method to get the exchange information from the exchange ID
	Fans	gethashArray	Method to get all the exchange IDs
	Initiator	recallCelebrity	To undo exchanges that have been initialized and not finalized by the buyer before the expiration
CelebrityNFTAuction	Auctioneer	initializeAuction	Initialize an open auction with the minimum and maximum bid amount and the bid duration
	Fans	getOpenBids	Method to get all the open bids
	Fans	getAuctionInfoByID	Method to return all the information about the auction from the Auction ID
	Bidder	bid	Perform a bid if the bid price is greater than the previous bid and the minimum bid amount
	Bidder	claimNFTByBidder	Claim the reward of the auction by the winning bidder of the auction
	Auctioneer	revokeAuction	Revoke the auction if no bids have been performed within the bid duration

5.2.2 CelebrityNFT Smart Contract

The CelebrityNFT smart contract defines the methods to purchase a random NFT and exchange NFTs between two fans. The smart contract also defines some of the utility methods that provide additional information for purchase and exchange transactions. We also define attributes that provide information related to the purchase price, percentage of rewards, and exchange information. The ExchangeInfo struct is used to store the exchange information such as the buyer, seller, celebrities' NFT being exchanged, price and expiry time. Figure 5.3 shows the smart contract's attributes, methods and structure. The code implementation of the smart contract is in Appendex A.2.

5.2.3 CelebrityNFTAuction Smart Contract

Figure 5.4 shows the atributes, methods, struct, and events used to create CelebrityNFTAuction smart contract. This smart contract is used to initialize auction, perform bids, and claim or revoke auctions. The smart contract uses AuctionInfo struct to store information such as seller, highest bidder, auctioned celebrity, maximum bid price, current bid price, expiry time, and active status of

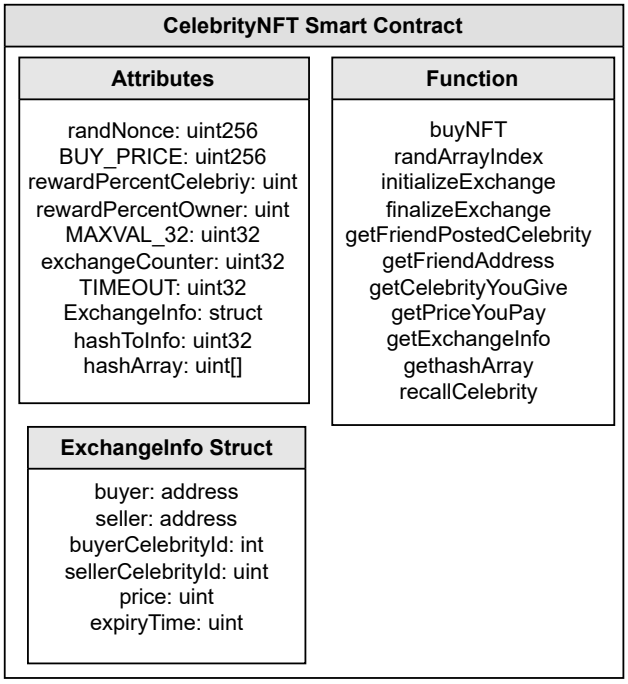


Figure 5.3: Smart Contract for purchase and exchange transactions

each auction. The event TransferSuccess is used to trigger an event notifying that the auction is complete. Appendex A.3 shows the code implementation of this smart contract.

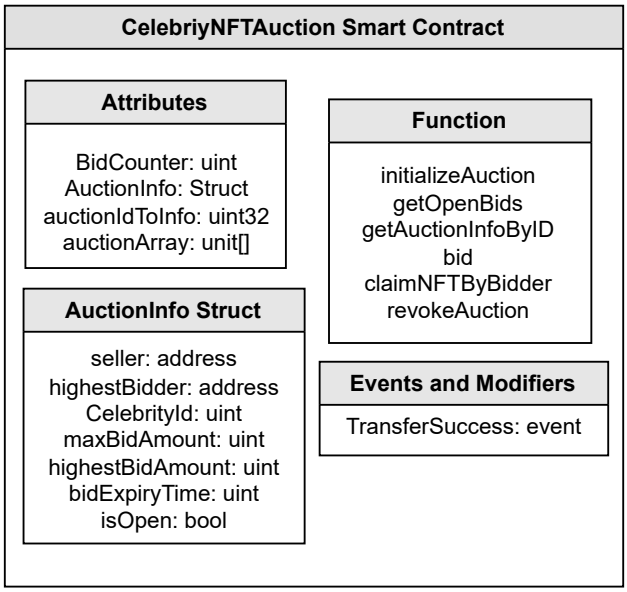
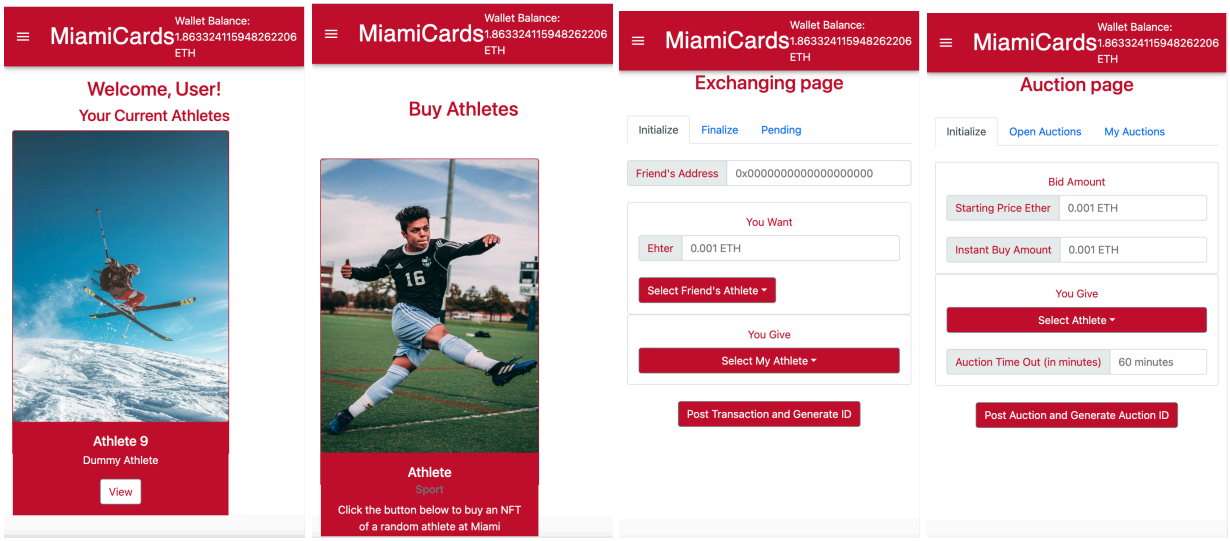


Figure 5.4: Smart Contract for auction transaction

5.3 User Interface



(a) Home page (b) Purchase page (c) Exchange page (d) Auction page

Figure 5.5: Screenshots of NNM Prototype

We create four web pages on NNM to their NFTs and perform purchases, exchanges, and auctions. Figure 5.5 shows screenshots of the homepage, buy page, exchange page, and auction page of NNM.

The home page in Figure 5.5a shows the wallet balance and the NFTs the fan owns (if any). For the purpose of this research, we have creat NFTs using stock images of celebrities taken from Splash [56], a popular website for stock images. Figure 5.5b shows the page for buying an NFT representing a random celebrity generated by the system.

The exchange page has three tabs: (1) 'Initialize' tab to initialize the exchange, (2) 'Finalize' tab to finalize the exchange (not displayed), and (3) 'Pending' tab to view pending exchanges (not displayed). Figure 5.5c shows the exchange page's 'Initialize' tab with text fields to input the wallet address of another fan, the amount in ETH to be received upon the completion of the exchange, and drop-down menus to select the celebrities that will be exchanged. The fan who initializes the exchange is called the initiator and the other fan is called the settler. All the initialized exchanges have a unique exchange ID. An initiator can view all of its initialized exchanges in the 'Pending' tab. All the expired transactions can be deleted by the initiator from the 'Pending' tab. The 'finalize' tab can be used by the settler to view the exchange posted by the initiator using the exchange ID and finalize it.

Figure 5.5d shows the auction page with the 'Initialize' tab. The 'Initialize' tab has the text fields to input the minimum and maximum bid amount, the bid timeout period, and the drop-down to select the card to initialize an auction. All the auctions submitted by the auctioneer can be viewed on the 'My Auctions' tab. The auctioneer can delete the auctions which have exceeded the bid timeout period from the 'My Auction' tab. All the auctions posted by other fans is visible on

the ‘Open Auction’ tab where bid can be performed.

5.4 Prototype Deployment & Experiment Setup

In this section, we describe the steps that we took to setup our experiment environment to evaluate the performance of NNM.

5.4.1 Deployment on PoW & PoS Test Networks

We deploy our smart contracts on both Ethereum and Avalanche test networks to perform performance comparison and analysis before deploying them on a suitable mainnet.

Ethereum previously used a proof-of-work consensus mechanism to validate transactions and create new blocks. On September 12, 2022, Ethereum transitioned from proof-of-work (PoW) to a more energy-efficient proof-of-stake (PoS) consensus mechanism [25]. To perform a comparison between PoW and the PoS consensus mechanism on Ethereum, we deployed NNM on the Rinkeby test network (now deprecated) which uses PoW, and on the Goerli test network which uses PoS. We used the free ETH that we received from Rinkeby Faucet and Goerli Faucet to perform our tests.

We also deployed NNM on the Avalanche Fuzi C-chain test network. We used free testnet tokens from Avalanche Faucet [57] to test our smart contracts.

5.4.2 Truffle as Web3 Development Environment

We use Truffle as NNM’s development environment and testing framework [58]. The development environment allows us to compile, test, deploy, and debug [59] NNM. The truffle tests are written in JavaScript which simulates the calls made to the contracts just like that done from the application’s front end, allowing for comprehensive testing [59].

5.4.3 Estimating Gas Price using Forge

We utilize Forge, a command-line tool [60], to estimate the gas prices (cost) for each API call in the smart contract. This estimation is important to understand the cost implications of executing the smart contract methods. We use the gas reports tool available on Forge to estimate the gas prices for individual contract calls. Forge supports the Solidity programming language, which enables us to write code for our experiment in Solidity.

5.4.4 Estimating Time Consumed by Each API Call using Truffle

To determine the time required (Latency) for each API call on NNM, we conduct an analysis using the truffle development environment [58]. The analysis is performed in JavaScript, where we measure the time before and after a function call and calculate the difference to determine the time delay for each API call

5.5 Evaluation & Results

In this section, we perform evaluation and analysis of NNM and discuss the results. We compare the cost of each API call on three different blockchain networks. Next, we test the scalability of our system by evaluating the gas consumed (GWEI) for different load overhead. Finally, we compare the latency of invoking smart contract methods on different blockchain networks employing different consensus mechanism.

5.5.1 Transaction Cost of Each API Call

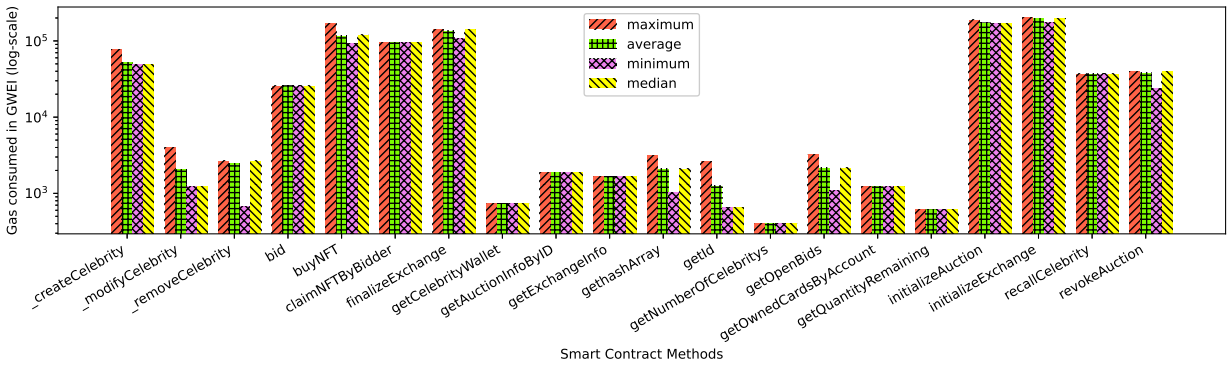


Figure 5.6: Gas estimate in GWEI (log-scale) for NNM contracts' API calls (10 calls each)

We perform our experiment to estimate the gas price for each NNM's smart contract method. Table 5.1 shows the list of smart contract methods along with their functions. Figure 5.6 shows the maximum, average, minimum, and median gas price of each smart contract method for 10 invocations. The methods that write to the blockchain network are bid, buyNFT, claimNFTByBidder, finalizeExchange, initializeExchange, initializeAuction, recallCelebrity, and revokeAuction. These are the methods that perform a gas-intensive task and cost much higher.

The methods _createCelebrity, _modifyCelebrity, and _removeCelebrity are the methods that do not update the blockchain state but modify local variables, resulting in lower gas consumption and lower cost compared to methods that update the blockchain. On the other hand, view methods like getCelebrityWallet, getAuctionInfoByID, getExchangeInfo, gethashArray, getId, getNumberOfCelebrity, getOpenBids, getOwnedCardsByAccount, getQuantityRemaining require the least amount of gas and costs least as they only access data from the blockchain without modifying it.

The table 5.2 shows the cost of executing each smart contract method on the Ethereum, Solana, and Avalanche blockchain networks. The cost of executing the smart contract methods is the least in the Avalanche network and highest in the Ethereum blockchain as per the prices taken on September 15, 2022.

Table 5.2: Average gas price in USD of NNM's smart contract API calls (average of 10 API calls) on 3 different blockchain networks – Ethereum, Solana, and Avalanche. The prices for Ethereum, Solana, and Avalanche are $1,507.83, $33.41, and $18.55 respectively on September 15, 2022 [2].

API	GWEI	Ethereum($)	Solana($)	Avalanche($)
_createCelebrity	51830	0.0781508289	0.0017316403	0.0009614465
_modifyCelebrity	2082	0.00313930206	0.00006955962	0.0000386211
_removeCelebrity	2475	0.00373187925	0.00008268975	0.00004591125
bid	25659	0.03868940997	0.00085726719	0.000475974
buyNFT	92988	0.140210096	0.00310672908	0.0017249274
claimNFTByBidder	92270	0.1391274741	0.0030827407	0.0017116085
finalizeExchange	120325	0.1814296448	0.00402005825	0.00223202875
getCelebrityWallet	741	0.00111730203	0.00002475681	0.00001374555
getAuctionInfoByID	1885	0.00284225955	0.00006297785	0.00003496675
getExchangeInfo	1678	0.00253013874	0.00005606198	0.0000311269
gethashArray	3266	0.00492457278	0.00010911706	0.0000605843
getId	1253	0.00188931099	0.00004186273	0.00002324315
getNumberOfCelebrities	583	0.00087906489	0.00001947803	0.00001081465
getOpenBids	2157	0.00325238931	0.00007206537	0.00004001235
getOwnedCardsByAccount	1698	0.00256029534	0.00005673018	0.0000314979
getQuantityRemaining	619	0.00093334677	0.00002068079	0.00001148245
initializeAuction	177949	0.2683168407	0.00594527609	0.00330095395
initializeExchange	176566	0.2662315118	0.00589907006	0.0032752993
recallCelebrity	24464	0.03688755312	0.00081734224	0.0004538072
revokeAuction	46052	0.06943858716	0.00153859732	0.0008542646

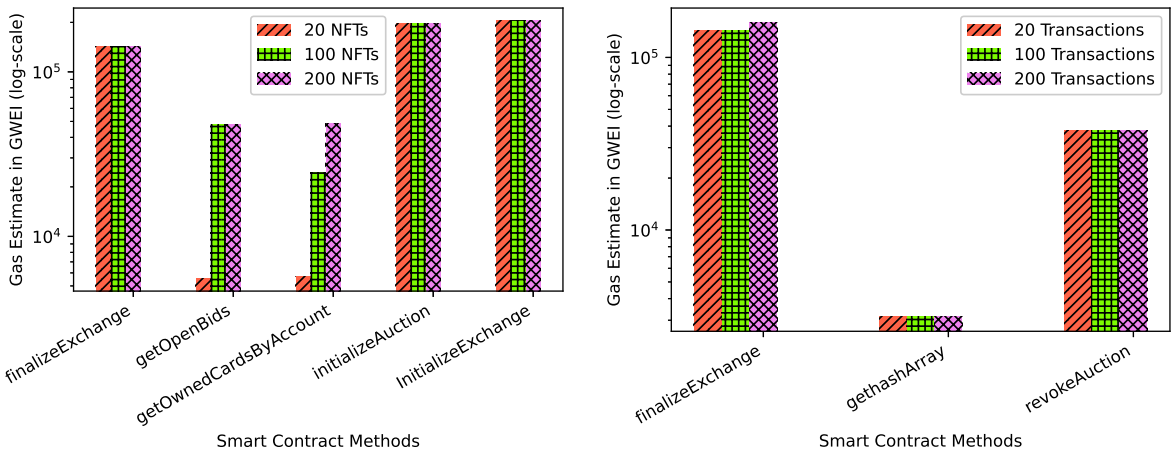
5.5.2 Checking Scalability of System with Varying Load

Subsequently, to understand scalability of NNM, we plotted the gas estimate in GWEI (log-scale) for different method invocations with varying load overhead in Figure 5.7. The legends represent the number of load overhead.

Figure 5.7a shows the gas estimates when the number of cards owned by fans participating in the smart contract invocation is different. It is clear from the figure that the methods finalizeExchange, initializeAuction, and initializeExchange are highly scalable as the gas estimate is not increasing with the increase in the load overhead. While the methods getOpenBids, and getOwnedCardsByAccount are not as scalable as other methods.

Figure 5.7b shows the gas estimate when the number of transactions in progress are different. We can see in the figure, the methods gethashArray and revokeAuction are highly scalable as the gas estimate does not vary with the varying load overhead. While finalizeExchange method is less scalable as the gas estimate increases with the increase in the number of load overhead.

From Figure 5.7, we found that most of methods are scalable while some methods are not and may create bottlenecks. However, the methods that are not scalable are the ones that are used less



(a) load overhead in terms of number of cards owned (b) load overhead in terms of number of transactions

Figure 5.7: Gas estimate in GWEI (log-scale) for NNM's smart contract method invocations when the load overhead varies. Load overhead is a storage overhead that may increase the gas estimate for an API invocation. Load overheads for different number of transactions are shown in the legend.

often and would not impact the overall performance of NNM.

5.5.3 Latency in Invoking Smart Contract Methods

Next, we analyzed the latency of smart contract method invocations on Rinkeby (PoW), Goerli(PoS), and Avalanche Fuzi C (PoS) chain blockchain testnet.. We used JavaScript code to measure the time taken by each method invocation and iterated it over 100 times. We plotted the time latency of each method on each of these test network which can be seen in Figure 5.8.

We can clearly see that the time consumed by the methods starting with *get* is less than that of any other method. It is because the methods beginning with *get* are view methods that do not make any changes to the blockchain because of which it takes less time to execute when compared to any other methods. Also, the methods _createCelebrity, _modifyCelebrity, and _removeCelebrity are the methods that only update the local variable and do not make any updates to the state of the blockchain. Because of this reason, these methods take less time to execute than the methods that update the blockchain network.

From the figure, it is evident that the time taken by API calls that update the blockchain's state is more than the methods that only view the blockchain state.

Upon comparing the time consumed by methods on the Rinkeby, Goerli blockchain, and Avalanche Fuzi C-chain blockchain, we can clearly see that the time delay for each API call is comparatively less on the Goerli and Avalanche testnet that use PoS consensus mechanism than that on the Rinkeby testnet which depends on PoW consensus mechanism.

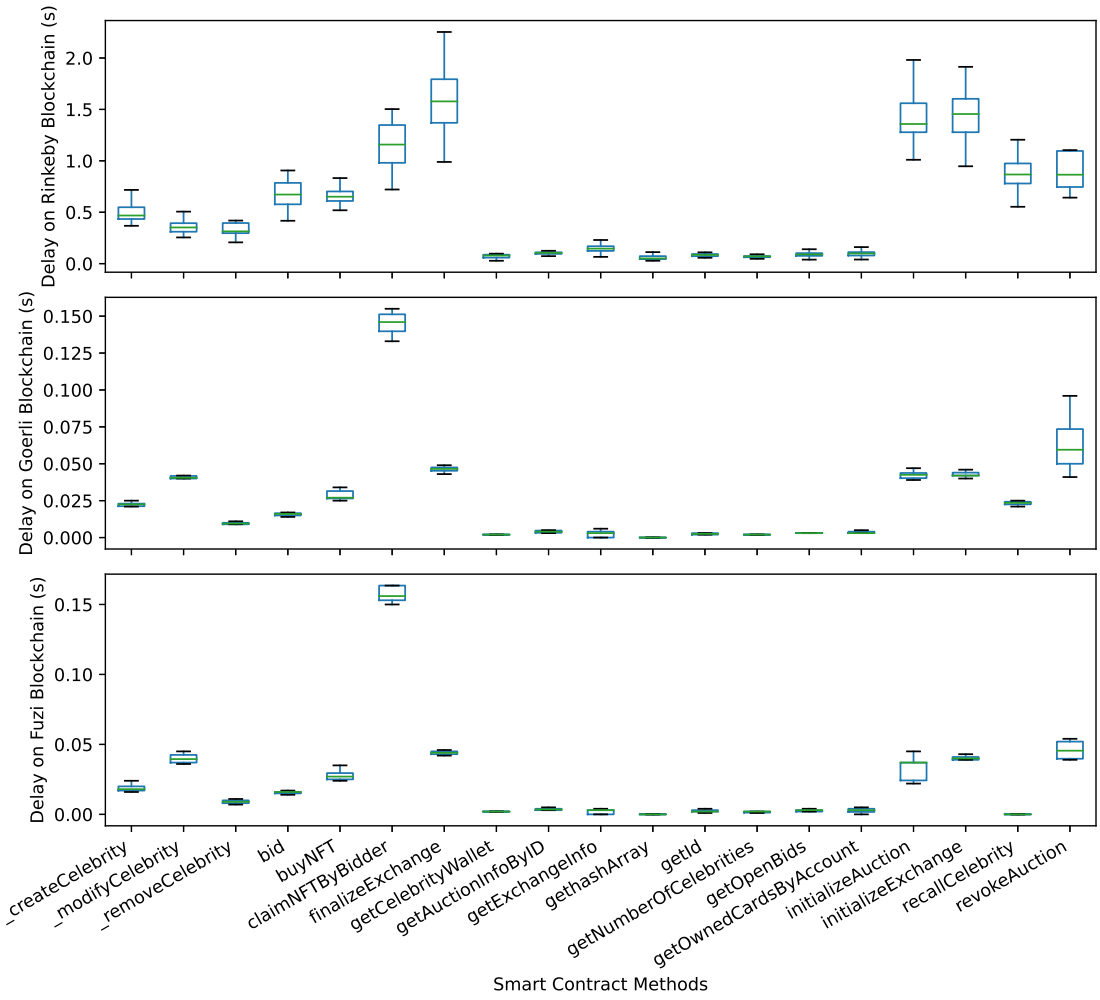


Figure 5.8: Latency in invoking smart contract methods on NNM

Chapter 6

System Demonstration and End User Feedback

In this section, we discuss the survey that we conducted with the end-user to evaluate the fairness and usability of NNM.

6.1 Survey Setup

The survey has been reviewed and approved by the Miami Research Ethics and Integrity Office, under protocol ID 04242e. Following the approval, undergraduate, and graduate students from the different Universities opted in as participants in response to email invitations. The eligibility criteria for users to participate in the study are as follows:-

- The user should be a minimum of 18 years of age.
- The user should have an undergraduate degree.

We had a total of 14 participants who volunteered to take part in a 30 minutes conversation. The survey was conducted in two different settings: online Zoom call and in-person meetup – as found appropriate by the participant. Out of 14 students, we had 11 graduate students and 3 undergraduate students. We had almost equal participation from both males and females: 8 males and 6 females.

In the survey, we also performed a 2 minutes demonstration of the application to demonstrate different features available on NNM. Following the demonstration, we asked the participants some questions related to fairness and usability of the application.

6.2 Questionnaires & Response

We asked 12 predefined questions to our participants and collected their feedback to perform analysis of fairness and usefulness of our solution. Table 6.1 summarizes the questionnaire and the responses. Here, questions numbered 1 and 2 are yes-no type, $3-7$ are descriptive questions related to fairness, $8-11$ are quantitative evaluations in the range of $[1-5]$ related to user experience, and 11 and 12 are open-response questions. Table 6.2 shows the individual responses of each interviewee to each of the first 7 questions.

From the end user feedback, we received significant input on ways we could improve the user experience and also make the application fair to the celebrities as well as the fans using it. From

Table 6.1: Interview Results from Potential Users(Fans)

Question	Questions	Response
Q1	Are you aware of blockchain-based applications? (yes/no)	3 of the participants answered with 'No's and 11 of them answered with 'Yes's.
Q2	Are you aware of the usage of Name Image Likeness (NIL) of student-celebrities? (yes/no)	11 of the participants answered with 'No's and 3 of them answered with 'Yes's.
Q3	Do you think there should be a fixed price for each NFT purchase or should it vary? (yes/no) Why?	11 of the participants answered that the purchase price should vary and 3 of them answered that the purchase price should be constant.
Q4	Do you think there should be a fixed proportion of money that should go to celebrities on each transaction? (yes/no) Why?	9 of the participants answered that the reward should be distributed in a fixed proportion (equality), 4 of them answered that the proportion of money/reward should be more for more popular celebrities (Aristotle's fairness concept), and 1 of them answered that the celebrity who makes fewer sales must get a higher reward (equity).
Q5	Do you think the users should be allowed to pick NFTs or should the NFTs be random during the purchase? (yes/no) Why?	9 of the participants answered that the users should be able to pick NFTs, 2 of them answered that the NFTs should be generated randomly, 1 answered that they should be given a random set to pick from, and 2 of them answered that they prefer both of the features.
Q6	Do you think enforced/regulated fairness is better than market-based fairness? (yes/no) Why?	7 of the participants answered that Enforced/Regulated fairness is better than market-based fairness, 4 of the participants answered that market-based fairness is better than enforced fairness, 2 of the participants answered that using both would be a better solution, and 1 of participants answered that it is not sure
Q7	Should the users be allowed to buy multiple NFTs in one click or one at a time? Why?	5 of the participants answered that the fans should be allowed to buy one NFT at a time, while 9 of the participants answered that the fans should be allowed to buy multiple NFTs at a time
Q8	How likely are you to use the software in the future? (1-5)	4 fans rated below 3, 7 fans rated 3, and 3 fans rated more than 3
Q9	How likely do you think the software/application will be of benefit to the users? (1-5)	2 fans rated 3, and 12 fans rated more than 3
Q10	How likely are you to trust the application with (security of) your money? (1-5)	2 fans rated below 3, 1 fans rated 3, and 11 fans rated more than 3
Q11	How likely are you to buy/trade NFTs (of your favorite celebrities) using the application? (1-5)	1 fans rated below 3, 4 fans rated 3, and 9 fans rated more than 3
Q12	What new features do you wish to be incorporated on the platform? (open response)	Allow to filters celebrities based on categories (e.g. categories of sports for celebrities), sorting and comparison of celebrities based on a given criteria, show celebrities who are registered, a community for networking, a concise process for performing transaction.
Q13	What new features do you wish to be incorporated on the platform? (open response)	A good concept to use blockchain but with few modifications on UI and features, the application can be of benefit to both the parties involved in a transaction.

the survey, we found that most of the participants are aware of the blockchain-based application because they have used it at one point in time or have heard about people using it. However, most of the participants did not have any idea about NIL.

6.3 Qualitative Fairness Analysis

When the fans were asked the fairness questions, the results were interesting. Most of the participants responded that the celebrities who are more popular should have a higher purchase price than others. This relates to the concept of fairness based on Aristotle's equality principle that says "*equals should be treated equally, and unequals unequally*"[61]. Basing their thoughts on the above

Table 6.2: Response to the Interview Questions by Individual Potential User/Fan

Interviewee	Q1	Q2	Q3	Q4	Q5	Q6	Q7
F1	No	No	Vary	Fixed	Pick	Regulated	One at a time
F2	Yes	Yes	Vary	Vary	Pick	Market	Multiple
F3	Yes	No	Vary	Vary	Both	Market	Multiple
F4	Yes	No	Vary	Fixed	Pick	Regulated	One at a time
F5	Yes	No	Vary	Fixed	Pick	Regulated	Multiple
F6	Yes	No	Fixed	Fixed	Pick	Regulated	One at a time
F7	Yes	No	Vary	Vary	Pick	Both	Multiple
F8	No	No	Vary	Vary	Pick	Regulated	One at a time
F9	Yes	No	Vary	Fixed	Random	Regulated	Multiple
F10	Yes	Yes	Fixed	Fixed	Random	Market	Any
F11	No	No	Vary	Fixed	Pick	Market	One at a time
F12	Yes	No	Vary	Fixed	Pick	Not sure	Both
F13	Yes	Yes	Vary	Fixed	Both	Market	Multiple
F14	Yes	No	Fixed	Vary	Random	Regulated	Multiple

response of Aristotle's equality principle, more than half of the participants also mentioned that the proportion of the reward each celebrity gets should be fixed since fixed percentage of higher amount is high and lower amount is low. However, their were concerns raised by the participants such as on what basis will the percentage be fixed.

On the other hand, participants who responded that the percentage should vary based their response on different logic. Some said that it should vary to motivate the players to perform better. One particularly interesting response among those who suggested a variable proportion points that the celebrities whose NFTs have less number of transactions should get more percentage than the ones who receive a higher number of transactions. This would balance the rewards earned by each celebrity and will help to achieve fairness in terms of having equal outcomes.

Interviewee F14 responded to fairness question number 5 saying:

> "*The proportion should vary and those who sell less should get more percentage and those who sell more should get less to maintain fairness.*"

When asked questions related to the user experience and fairness from fans' perspective, we found that what is fair to the fans has a higher chance of not being fair to the celebrities. For example, most of the participants responded that the fans should be allowed to pick the NFTs of their favorite celebrities. Only a few responded that the NFTs should be generated randomly. One of the participants' responses was that they should be allowed to pick from a random set so that it is fair in both ways.

But interestingly, when talking about market-based fairness and regulated fairness in question number 6, most of the participants responded that fairness should be regulated to avoid any unpredictable outcomes and instability in the market. Some also reasoned stating that enforced fairness ensures that there is not any kind of market manipulation or misuse. One of them gave a beautiful

reason why the market should be regulated. The reason was that market-based fairness can lead to an unfair advantage to some and an unfair disadvantage to others; however, in regulated fairness, there is a cap to that. Interviewee F14 responded to Q6 saying:

> "*It should be enforced fairness. In market-based fairness, it can be an unfair advantage for some and an unfair disadvantage for others. Enforced fairness will have a cap* [on the advantages and disadvantages.]"

While some contradicted this by supporting market-based fairness by providing variety of reasons. Some reasoned that regulated fairness creates a centralized control which is not what blockchain technology is about. Some of the participants also said that having some level of both market-based and enforced fairness would be more beneficial from both fans' and celebrities' perspectives. Participant F11 states:

> "*I think it should be market-based fairness from the users' perspective and regulated fairness from the celebrities' perspective.*"

When the candidates were asked about their perspective on Q6, the response suggests that majority of the fans prefer being able to buy multiple NFTs at a time. As they reasoned it would consume less gas and would reduce the transaction cost rather than when bought separately one-at-a-time. While some suggested one-at-a-time with strong reason such as allowing to pick card one-at-a-time allows the fans with time to think and see how the celebrities are performing over time. Another participant mentioned that one-at-a-time is better as it makes sure that the market is not exhausted and there is still variety of NFTs in the market for others to purchase.

6.4 Quantitative Usability Analysis

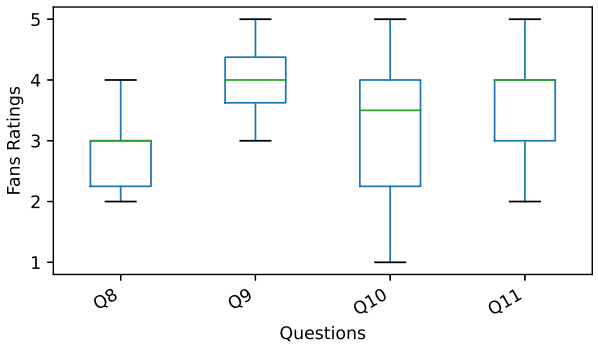


Figure 6.1: Ratings given by fans with 5 being the highest and 1 being the lowest

Figure 6.1 shows the quantitative evaluation of NNM in perspective of user satisfaction. Participants rated the solution with a median value 3 in terms of how likely they would use the application in the future. While, when they were asked to rate when they were allowed to trade NFTs of their

favourite celebrities, the median rate is 4. When rating the solution based on how useful the application will be of benefit to the actual users, most of them rates it to be 4. Also, upon asking to rate how much they trust the blockchain network with the security of their money, most of them rated with 4.

Based on the quantitative user satisfaction analysis, we found that the fans are more likely to use the application when they are able to buy/trade an NFT of their favorite celebrities.

6.5 Open-Response Analysis

Lastly, from the open-response questions, the participants pointed concerns realated to UI and the ease of use. They suggested that the UI should be made much better and easier to understand.

> "*The process from the beginning to the end [of a transaction] should be [kept] simple and short.*"

In addition, the fans also suggested that they would like to have a community platform for interaction. Also, some fans suggested that they should be able to filter celebrities based on categories (for example categories of sports for athletes), and should be able to sort and compare celebrities based on certain criteria. Also, they should be able to see the celebrities that are registered on NNM and also be allowed to select celebrities of their choice for purchase with an additional price than when purchased randomly generated NFT.

Some of the feedback related to the technology that we received was that the concept of using blockchain for building an application is a good concept as blockchain is on the rise. Also, they found the idea interesting and would be happy to see more enhancements to the existing prototype.

6.6 Summary of End-User Feedback

Some of the significant insights from the interview summary in table 6.1 are listed below.

1. More than half of the participants responded that the fans should be allowed to pick the NFT of their own choice.
2. More than half of the participants responded that the purchase price should vary based on the celebrity.
3. Half of the participants responded that the fans should be allowed to buy multiple NFTs at a time.
4. More than half of the participants responded that the reward percentage should be fixed.
5. Half of the participants responded that fairness should be enforced.

Chapter 7

Enforcing Fairness in Total rewards Earned by Celebrities

As discussed in Sections 3.3.2 and 4, we aim to provide reasonable fairness in total rewards earned by celebrities. Celebrities earn rewards in two main markets: 1) when fans purchase an NFT in the primary market, and, 2) when fans exchange, auction, or sell NFTs by themselves in the secondary market. The smart contract only has control over the primary market where the NFTs are randomly selected during purchase. In the secondary market, the price of each NFT is decided by fans and is mostly proportional to the popularity of the celebrity [62]. In this section, we describe how fairness could be enforced by performing a correction on the probability distribution of celebrity selection in the primary market when NFTs are purchased.

7.1 Fairness in Earned Rewards

To achieve distributive fairness in NNM, we aim to ensure a fair distribution of rewards among celebrities. As per [46] and [63], achieving distributive fairness requires a mechanism for fair and impartial allocation of resources and effective conflict management. To implement this in NNM, we have used the traditional concept of "cake cutting" [64], where the cake represents the rewards earned by the celebrities, and our goal is to ensure a fair division of these rewards.

The distribution of the rewards can be achieved in two ways: egalitarian and envy-free approaches.

1. **Egalitarian** means that everyone should be treated equally. This implies that all celebrities should get equal distribution of rewards regardless of their popularity. From the case study conducted on student-athletes, we found some of them in favor of egalitarian fairness. For example:

   > "*While you have the time, while you're doing it at that level like, you should absolutely be able to make the money you know, like everyone else in every other industry gets to do.*"

2. **Envy-free** means that everyone should receive a fair share of rewards. This implies that each celebrity receives the share that they deserve and no more than their own share. While some of the student-athletes from the case study supported egalitarian fairness, the rest of them thought that envy-free way of being rewarded is fair for celebrities of different popularity and likeness. For example:

"*There are definitely certain athletes — I'm mostly pertaining to grossing sports like football and basketball, who, I believe, really do deserve to make some sort of profit.*"

From the interviews conducted with student-athletes as discussed in Section 3.1, we found both kinds of rewards sharing as desirable i.e. everyone should get the same rewards, and more popular celebrities should get more rewards. Achieving egalitarian rewards earning is a bit difficult as it requires the system to enforce the rewards earned by each celebrity. In the subsequent section, we describe intervention techniques that can enforce egalitarian fairness by adjusting the probability of selecting a celebrity for NFT purchase in the primary market, as we do not have any control over the transactions performed in the secondary market.

7.2 Providing Envy-free Fairness

In the primary market, all the celebrities earn the same fixed rewards (a fraction of the NFT purchase price). The fans/users dictate the exchange and auction price in the secondary market, which is proportional to the popularity of the celebrity, thus achieving envy-free rewards earnings. In NNM, the probability distribution for picking and associating a celebrity to an NFT during NFT purchase follows a uniform distribution. Here probability of picking a celebrity, c_i is

$$P(c_i) = \frac{1}{N}$$

where, N is the total number of celebrities. As fans purchase NFTS, each celebrity has an equal chance of getting picked, and as the number of NFTs gets bigger in the system, the difference in the number of cards associated with celebrities gets less. Since the number of cards associated with celebrities is almost equal, the total rewards from the secondary market is solely dependent on the auction price, which is decided by the fans. Considering fans want to procure celebrity NFT with higher popularity, the rewards earned from the secondary market is proportional to the performance of the celebrity, entailing envy-free fairness.

7.3 Egalitarian Fairness by Correcting Probability in Primary Market

The price of NFTs once purchased from the primary market cannot be controlled on the secondary market. NNM has only control over the probability distribution of associating celebrities to an NFT during NFT purchase in the primary market. Maintaining uniform distribution results in envy-free rewards fairness. We propose an algorithmic approach for enforcing egalitarian fairness by adjusting the probability distribution periodically at every correction interval (`CI`). The probabilities are adjusted to reduce the number of new NFTs for high-earning celebrities.

First we check the earned rewards for each celebrity $(\pi(c))$ and determine a unnormalized probability distribution function as

$$P'(c) = \left(1 - \frac{\pi(c)}{\sum_{c \in \mathbb{C}} \pi(c)}\right)^{\texttt{PCE}} \tag{7.1}$$

where $(\texttt{PCE})$ is the probability correction parameter which is a weight that influences how fast the probabilities converge.

Finally, normalize the probability distribution function. Thus, the probability distribution function of associating a celebrity with a newly purchased NFT is

$$P(c) = \frac{P'(c)}{\sum_{c \in \mathbb{C}} P'(c)} \tag{7.2}$$

Where $\mathbb{C}$ is the set of all celebrities in the system, this correction in the probability distribution in the primary market enforces the probability of newly generated NFTs to be inversely proportional to the total rewards earned by each celebrity. As the number of cards is limited for high-earning celebrities, the total earned rewards converges to an equal amount, as we can observe in the later sections.

7.4 Modeling Rewards Earned by Celebrities

Correctly modeling the price of NFTs in the secondary market is very difficult as it involves so many factors such as popularity, scarcity of NFTs, and human emotion. We simplify the model and consider that the price of NFTs in secondary market is dependent only on the popularity of the celebrity that the NFT represents. Higher popularity would imply a high price and lower popularity would imply a lower price. We consider that the celebrities could be arranged in ascending order of their popularity and number them $[1 \ldots N]$ where N is the number of celebrities. We model the price of each NFT $(\texttt{NFT}_\texttt{p})$ in the secondary market as

$$\texttt{NFT}_\texttt{p} = \texttt{PP} + U(0, \texttt{PV}\left(\frac{\theta_c + N}{N}\right)^{\texttt{RPE}}) \tag{7.3}$$

where, $\texttt{PP}$ is the base purchase price, $\texttt{PV}$ is the variable price, and θ_c is the popularity of the celebrity c featured on the NFT. We used rewards popularity exponent $(\texttt{RPE})$ which differentiates the average price based on the popularity of a celebrity. Higher the value of $\texttt{RPE}$, the more price difference among the celebrities based on their popularity. $\texttt{RPE}$ can have different values. $\texttt{RPE} = 0$ means the average price for any celebrity does not change with respect to their popularity. In that setup, no correction would be required. $\texttt{RPE} = 1$ represents that the price of each NFT is linearly affected by the popularity while $\texttt{RPE} = 2$ represents that the price of each NFT is exponentially affected by the popularity and so on.

Another way the price could be modeled is

$$\texttt{NFT}_\texttt{p} = \texttt{PP} + U(0, \texttt{PV}(\theta_c^{\texttt{RPE}})) \tag{7.4}$$

where we allow the price of the NFTs to be extremely high.

Next, we model the amount of rewards earned by each celebrity upon transactions of NFTs in both primary and secondary markets. Let m_p^c represent the total number of purchases for a celebrity (c) in the primary market. The total amount of rewards earned by c from the purchases of $\texttt{NFT}_c$ is:

$$\pi_p^c = m_p^c \texttt{PP} \epsilon_c \tag{7.5}$$

If there are m_e^c number of total exchanges for a celebrity (c), the total amount of rewards earned by c from all the exchanges of $\texttt{NFT}_c$ is:

$$\pi_e^c = \sum_{j=1}^{m_e^c} \pi_{e_j}^c \tag{7.6}$$

If there are m_a^c number of total successful auctions of NFTs representing celebrity (c), the total amount of rewards earned by c from auctions of is:

$$\pi_a^c = \sum_{j=1}^{m_a^c} \pi_{a_j}^c \tag{7.7}$$

The total rewards earned by the celebrity (c) from all the transactions performed on $\texttt{NFT}_c$ is:

$$\pi^c = \pi_p^c + \pi_e^c + \pi_a^c \tag{7.8}$$

7.5 Simulation Setup

For investigating the rewards earned by each celebrity upon performing some transactions, we performed a simulation using the fairness model discussed in Section 7.4, and probability correction in Section 7.3. We performed the simulation for 10 million transactions and 100 celebrities. We randomly generated the popularity index for each celebrity within a range $(0,\ N)$ where 0 is the lowest popularity and N is the number of celebrities (i.e. 100) which is the highest popularity. Initially, we set the probability of selecting each celebrity to be equal to $1/N$.

We took the simulation result to find the probability distribution of generating each of the 100 celebrities after performing $10M$ transactions on them. We also calculated total number of NFTs generated for each celebrity and total amount of rewards earned by each celebrity after performing $10M$ transactions. To compare among variety of situations, we performed the simulation for different values of $\texttt{CI}$, $\texttt{RPE}$, and $\texttt{PCE}$. Each CI is a total of 10 purchases and 1 random transactions (either a purchase, exchange, or auction). That is, for $\texttt{CI} = 1$, we perform correction only after performing 10 purchases and 1 random transaction of any type.

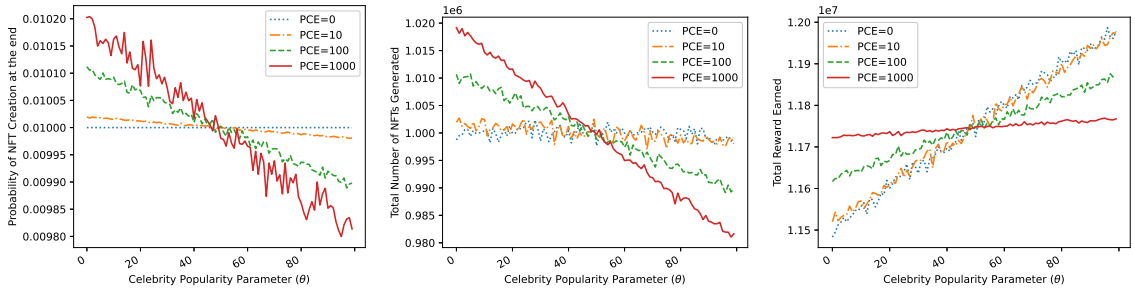
To perform analysis and view the results, we plotted the data for probability distribution, number of NFTs and total rewards earned in the Y-axis and the celebrities (ordered in terms of popularity) in the X-axis. Finally, to evaluate the fairness for NNM, we took the total rewards earned by each celebrity over 10 million transactions and observed the Jain's index for different values of $\texttt{CI}$, $\texttt{RPE}$,

and PCE.

We used Python programming language [65] to write our simulation code. We used the Python random module [66] to pseudo-randomly select a celebrity representing an NFT for a transaction. We used Matplotlib [67], python data visualization library, and Pandas [68] python data analysis libraries for plotting our graph.

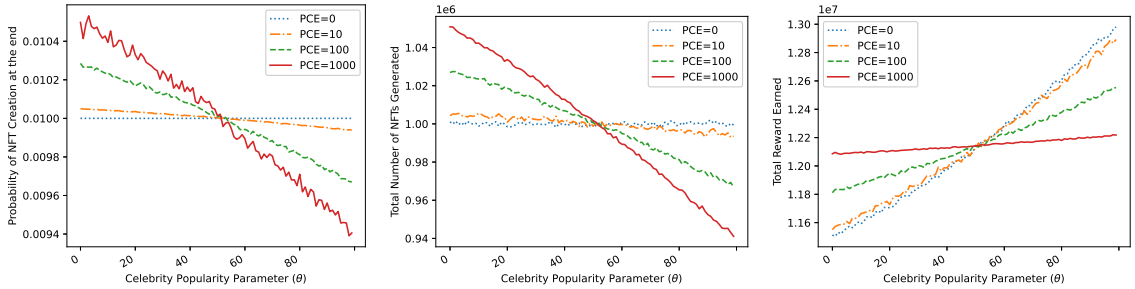
In the rest of the section, we describe the effect of PCE, RPE, and price in the secondary market on the total rewards earned by each celebrity.

7.5.1 Observing Fairness for Correction Interval, $\texttt{CI} = 1$



(a) Probabilities (b) Number of NFTs (c) Total rewards

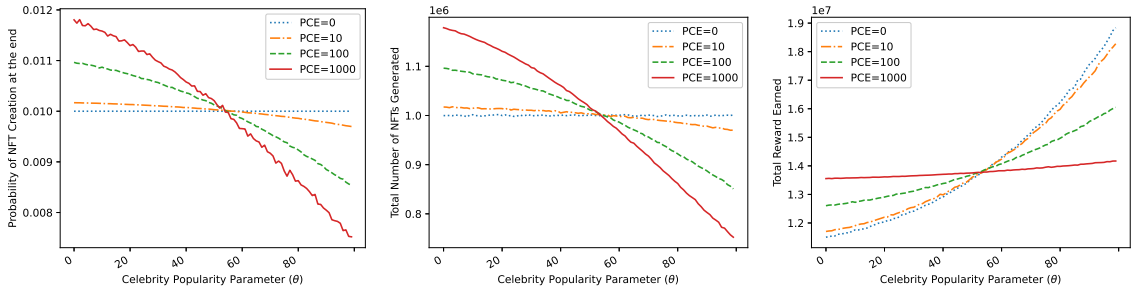
Figure 7.1: Simulation result for $10M$ transactions, $\texttt{RPE} = 1$ and $\texttt{CI} = 1$



(a) Probabilities (b) Number of NFTs (c) Total rewards

Figure 7.2: Simulation result for $10M$ transactions, $\texttt{RPE} = 2$ and $\texttt{CI} = 1$

Figures 7.1, 7.2, 7.3 show the plots for the probability distribution, number of total NFTs generated and total rewards earned by each celebrity with respect to the celebrity popularity parameter (θ). The simulations are done for correction interval of $\texttt{CI} = 1$ and the rewards popularity exponent $(\texttt{RPE})$ equals 1, 2, and 4. We plot each of the dependent variables in the Y-axis for different values of PCE shown in the legends. From each of these figures, we see that the resulting plot is different for the different values of PCE. When $\texttt{PCE} = 0$, the probability distribution and number of NFTs generated remains about the same for each celebrity but the rewards earned varies drastically. However, when observing the plots for $\texttt{PCE} = 1000$, the probability distribution and the number of



(a) Probabilities (b) Number of NFTs (c) Total rewards

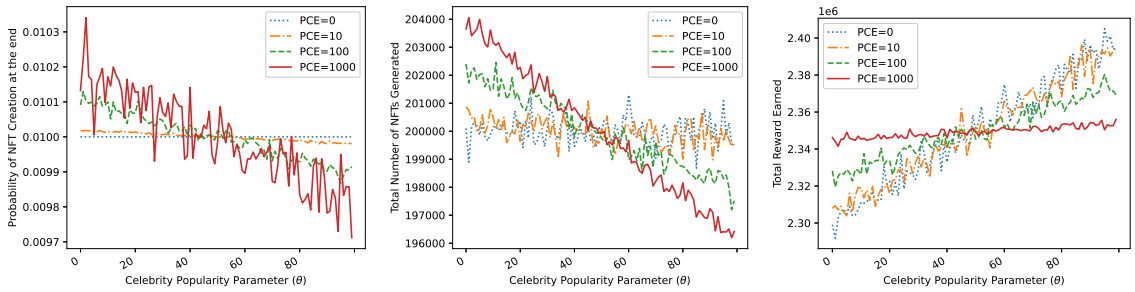
Figure 7.3: Simulation result for $10M$ transactions, $\texttt{RPE} = 4$ and $\texttt{CI} = 1$

NFTs is different, with higher probability for loss popularity celebrities and lower probability for higher popularity celebrities. But, for $\texttt{PCE} = 1000$, each celebrity is able to benefit equally from the transactions. Hence, from this we conclude that higher value of PCE leads to a fair distribution of rewards among multiple celebrities.

In Figures 7.1c, 7.2c, and 7.3c, the average of total rewards earned by each celebrity is more for $\texttt{RPE} = 4$ than for $\texttt{RPE} = 1$. From this comparison, we conclude that the celebrities can benefit more from a higher value of RPE.

We can thus infer that with higher values of PCE and RPE, we are able to achieve a fair distribution of rewards among all the celebrities.

7.5.2 Fairness for Correction Interval, $\texttt{CI} = 5$

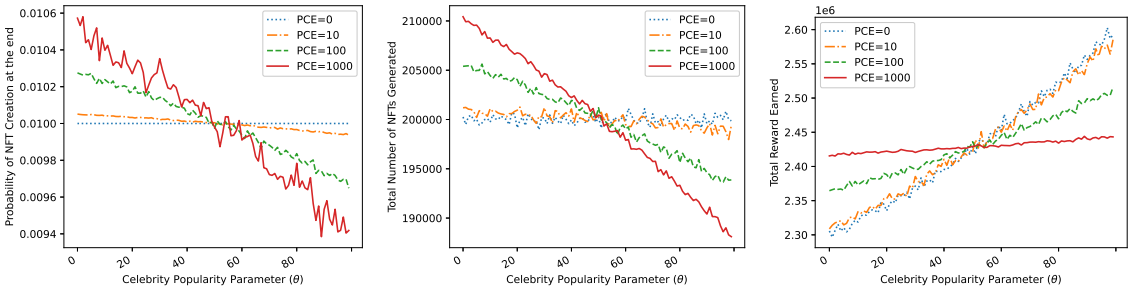


(a) Probabilities (b) Number of NFTs (c) Total rewards

Figure 7.4: Simulation result for $10M$ transactions, $\texttt{RPE} = 1$ and $\texttt{CI} = 5$

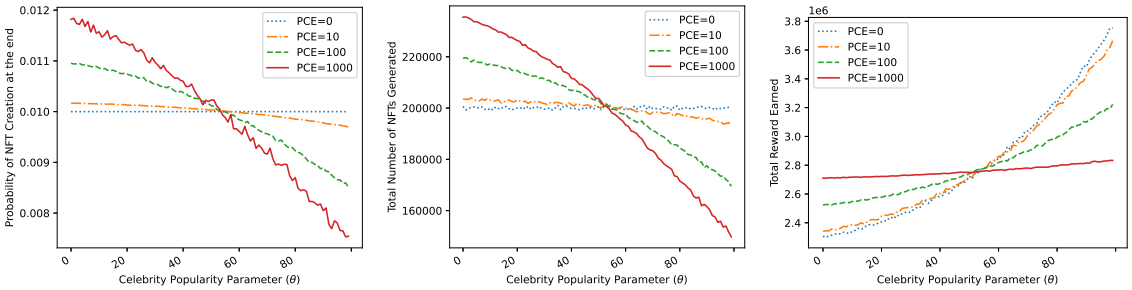
Figures 7.4, 7.5, and 7.6 shows the plot for $\texttt{CI} = 5$ and RPE equals to 1, 2, and 4.

Upon comparing the simulation result with $\texttt{CI} = 1$ in Figures 7.1, 7.2, and 7.3, and that with $\texttt{CI} = 5$ in Figure 7.4, 7.5, and 7.6, the former is more smooth when compared to the later. This implies that the total rewards earned is deviating more from the least square regression line (i.e. have high standard deviation) for $\texttt{CI} = 5$. However, the difference is insignificant when we compare the total rewards earned when $\texttt{RPE} = 4$ and $\texttt{PCE} = 1000$ for both $\texttt{CI} = 1$ and $\texttt{CI} = 5$.



(a) Probabilities (b) Number of NFTs (c) Total rewards

Figure 7.5: Simulation result for $10M$ transactions, $\texttt{RPE} = 2$ and $\texttt{CI} = 5$



(a) Probabilities (b) Number of NFTs (c) Total rewards

Figure 7.6: Simulation result for $10M$ transactions, $\texttt{RPE} = 4$ and $\texttt{CI} = 5$

We also observe that the average of total rewards earned when $\texttt{CI} = 5$ is less (approximately $2.8e6$) than when $\texttt{CI} = 1$ (approximately $1.4e7$) with $\texttt{RPE} = 4$ and $\texttt{PCE} = 1000$. Hence, from this comparison, we conclude that the performing frequent corrections to adjust the probability of generating NFTs leads to a fair reward distribution with higher amount.

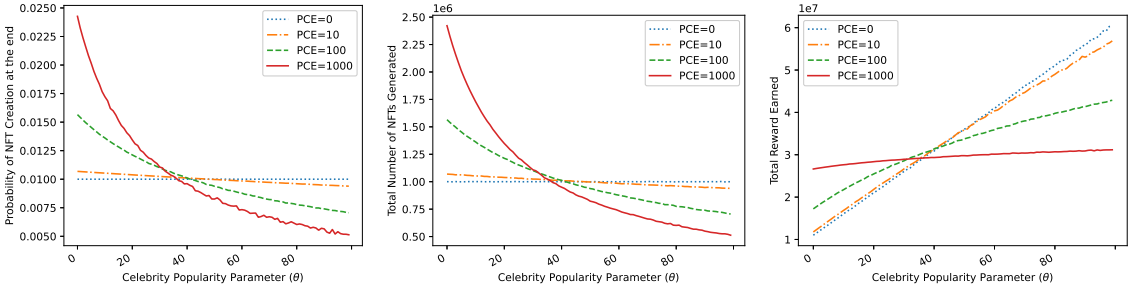
7.5.3 Fairness for Unrestricted Price in Secondary Market

In the earlier section, we limit the price of NFTs in secondary market as in equation 7.3. In this section, we conduct fairness analysis in rewards distribution when the price of NFTs is not limited as in equation 7.4

In Figures 7.7, 7.8, and 7.9, we plot the data obtained from performing the simulation without restricting the price of the NFT in the secondary market. We use the Equation 7.4 to perform the simulation for different values of PCE and RPE with $\texttt{CI} = 1$.

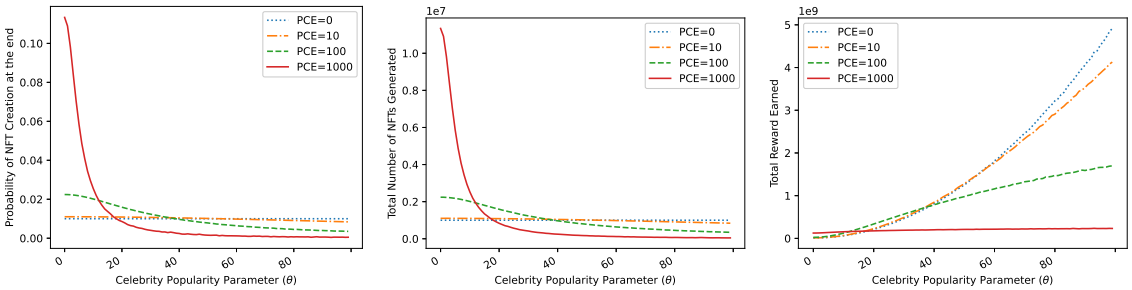
From the plots, we can see the increasing gap between the lowest and highest popularity (when we look for $\texttt{PCE} = 0, 10, 100$) with the reward varying in the range of $10e13$ for $\texttt{RPE} = 4$. Also, the number of NFTs that are generated for lower-popularity celebrities is huge when compared to high-popularity celebrities for $\texttt{PCE} = 0, 10, 100$.

However, even when the price is unrestricted, the total rewards earned when $\texttt{PCE} = 1000$ has an equal distribution. This implies that our correction algorithm is robust even with the unpredictabil-



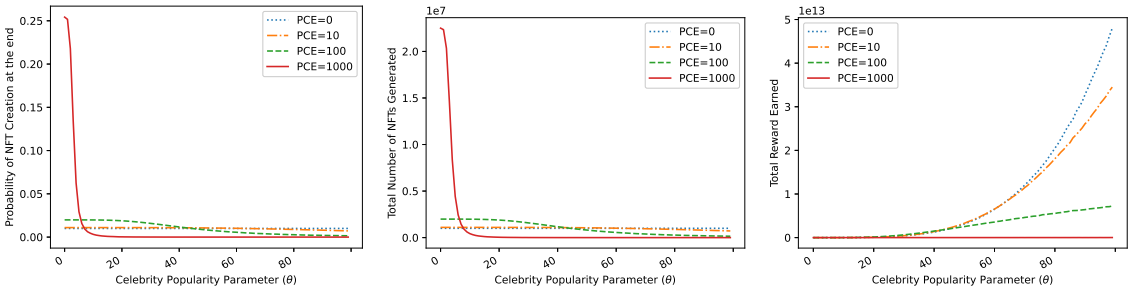
(a) Probabilities (b) Number of NFTs (c) Total rewards

Figure 7.7: Simulation result for $10M$ transactions, $\texttt{RPE} = 1$, $\texttt{CI} = 1$ with unrestricted price in secondary market



(a) Probabilities (b) Number of NFTs (c) Total rewards

Figure 7.8: Simulation result for $10M$ transactions, $\texttt{RPE} = 2$, $\texttt{CI} = 1$ with unrestricted price in secondary market



(a) Probabilities (b) Number of NFTs (c) Total rewards

Figure 7.9: Simulation result for $10M$ transactions, $\texttt{RPE} = 4$, $\texttt{CI} = 1$ with unrestricted price in secondary market

ity in the price of NFTs in the real-world secondary market.

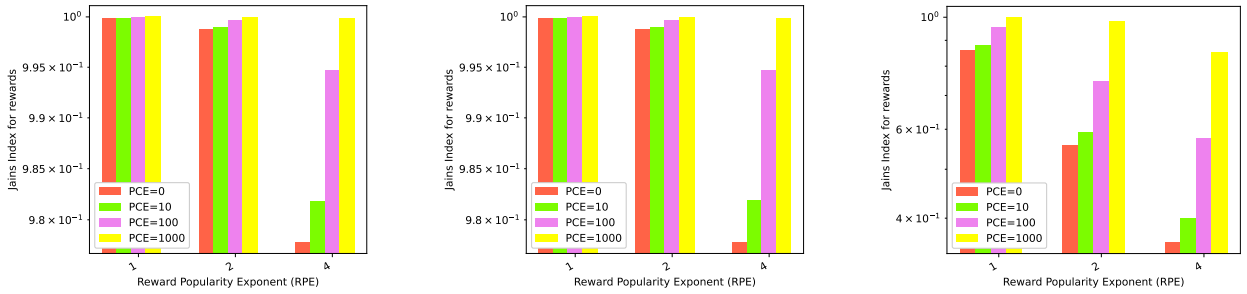
7.6 Measure of Fairness using Jain's Index

Fairness is a qualitative measure, and there is no strong measurement of fairness that we can directly apply in this situation. We use Jain's fairness index [69], which is applied in network science, to evaluate the fairness of rewards earned by celebrities. In this index, fairness could be measured as:

$$FI = \frac{(\sum \pi^c)^2}{n \sum (\pi^c)^2} \tag{7.9}$$

where π^c is the total rewards earned by a celebrity (c). The value of Jain's index range from 0 to 1. 0 represents that the system is not fair, while 1 represents that the system is extremely fair (i.e. all celebrities earn equally).

We used Equation 7.9 to calculate the fairness index for total rewards earned by each of the 100 celebrities over 10 million transactions for different values of `CI`, `RPE`, and `PCE`.



(a) `CI` $= 1$ (b) `CI` $= 5$ (c) Unrestricted price for `CI` $= 1$

Figure 7.10: Total rewards

Figures 7.10 shows the plot of fairness indices for total rewards earned. From the figure, we can clearly see that the fairness index is higher for higher value of `PCE`. Also, it is clear that for `RPE` $= 1$, the Jain's fairness index is the highest. While for `RPE` $= 4$, the fairness index declines. Hence, the most suitable value of `PCE` and `RPE` which maximizes the Jain's index for total rewards is 1000 and 1 respectively. That implies that the higher the value of `PCE` and `RPE` the better the result. However, when comparing Jain's index between Figure 7.10a and Figure 7.10b, there is no significant difference.

In case of Jain's index for total rewards when the price at secondary market is not restricted shown in Figure 7.10c has comparatively less fairness index when compared with Figures 7.10a and 7.10b.

Chapter 8

Conclusion

During our research, we found that there is a clear lack of a digital marketplace for non-fungible tokens (NFTs) that offers transparency, fairness, and rewards celebrities in the secondary market. To address this gap, we adopted a design science approach and conducted a survey with student-athletes as a representative group of celebrities. Our aim was to understand the motivation behind creating a fair NFT trading marketplace that treats each celebrity equally and provides them with a fair reward for their collectible exchanges. Based on our findings, we developed NNM, a cost-effective and highly scalable marketplace for trading digital collectibles that represent the name, image, and likeness (NIL) of celebrities. Our platform aims to achieve two types of fairness. Firstly, envy-free fairness is automatically ensured in the secondary market where the price of NFTs is determined by the fans themselves. Secondly, we promote egalitarian fairness by enabling a uniform distribution of all celebrities in the NFTs during the primary market purchase. Finally, we successfully implemented NNM and conducted demonstrations to a group of users through semi-structured interviews. The feedback from these interviews indicated that NNM is perceived as fair and useful by the participants.

Threats to Validity

There are some assumptions that we made during the research that may lead to threats to validity. Some of the threats to validity are described below:

- Our research focused exclusively on the perspective of one particular group of celebrities, namely student-athletes, in order to establish our goals. However, it is important to acknowledge that this approach may have limitations, as it did not take into account other groups of celebrities that may differ significantly from student-athletes.
- Our model solely incorporates popularity as a determining factor in influencing the price of each celebrity. However, it is important to recognize that this assumption is incomplete since, in reality, the price of each collectible is influenced by various circumstances and factors.
- In our simulation, we performed more number of purchases that trades in secondary market. Our correction interval $(\texttt{CI})$ had 10 purchases but only 1 random transaction. However, in real scenario, it might be a completely opposite case. This may significantly impact the conclusions drawn.
- With the current correction model, NFTs of less popular celebrities will be flooded in the marketplace making them even less valuable while there could only be 1 NFT of a highly popular celebrity which could cost millions in secondary market.

Future work

- In our existing model, when conducting an NFT exchange without any monetary involvement, the celebrities do not receive any rewards. In the future, we need to explore methods to ensure a continuous allocation of rewards to the celebrities in such exchange transactions.
- Based on the end-user case study, most of the participants mentioned that they would prefer to pick NFTs of their own choice. So, future works could involve a thorough investigation on ways maintain fairness and to assign price to NFTs available to choose rather than going through a randomized deck of NFT.
- In future, we can implement scenarios with transactions from that resemble the real world pattern to evaluate the fairness of the system.

References

[1] Mohammad Jabed Morshed Chowdhury, Alan Colman, Muhammad Ashad Kabir, Jun Han, and Paul Sarda. Blockchain versus database: a critical analysis. In *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 1348–1353. IEEE, 2018.

[2] Cryptocurrency prices, charts, and market capitalizations. `https://coinmarketcap.com/`.

[3] Zibin Zheng, Shaoan Xie, Hong-Ning Dai, Xiangping Chen, and Huaimin Wang. Blockchain challenges and opportunities: A survey. *International Journal of Web and Grid Services*, 14(4):352–375, 2018.

[4] Sonam Adinolf and Selen Turkay. Collection, creation and community: a discussion on collectible card games. In *Proceedings of the 7th international conference on Games+ Learning+ Society Conference*, pages 3–11, 2011.

[5] Trading card. `https://en.wikipedia.org/wiki/Trading_card`, Oct 2022.

[6] Name image likeness. `https://www.firstpointusa.com/name-image-likeness/`.

[7] Name image likeness guidelines. `https://gowilkesu.com/sports/2021/8/4/name-image-likeness-guidelines.aspx`.

[8] Braly Keller. What is name, image, and likeness? `https://opendorse.com/blog/what-is-name-image-and-likeness/`, Apr 2021.

[9] Mike Orcutt. This blockchain-based card game shows us the future of ownership. `https://www.technologyreview.com/2019/07/11/693/this-blockchain-based-card-game-shows-us-the-future-of-ownership/`, Apr 2020.

[10] How blockchain is powering digital collectibles for the sports industry. `https://www.protokol.com/insights/how-blockchain-is-powering-digital-collectibles-for-the-sports-industry/`, Jan 2022.

[11] Andreas M Antonopoulos and Gavin Wood. *Mastering ethereum: building smart contracts and dapps*. O'reilly Media, 2018.

[12] Jake Frankenfield. What is ethereum? https://www.investopedia.com/terms/e/ethereum.asp, Jun 2022.

[13] 11 ways ethereum can benefit enterprise: Consensys. https://consensys.net/enterprise-ethereum/best-blockchain-for-business/11-ways-ethereum-can-benefit-enterprise/.

[14] Marcus Soulsby. The benefits of the ethereum blockchain. https://medium.com/plutus/the-benefits-of-the-ethereum-blockchain-f332e62f7659, Sep 2019.

[15] The value of uniqueness: Non-fungible tokens in the age of name, image and likeness. https://www.natlawreview.com/article/value-uniqueness-non-fungible-tokens-age-name-image-and-likeness, Jul 2021.

[16] All about nfts. https://future.a16z.com/podcasts/nfts-explainer/, Nov 2021.

[17] Dominic Chalmers, Christian Fisch, Russell Matthews, William Quinn, and Jan Recker. Beyond the bubble: Will nfts and digital proof of ownership empower creative industry entrepreneurs? *Journal of Business Venturing Insights*, 17:e00309, 2022.

[18] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*, page 21260, 2008.

[19] Pauline Adam-Kalfon, Selsabila El Moutaouakil, and C Richard. Blockchain, a catalyst for new approaches in insurance. *Paris: PwC https://www. pwc. com/gx/en/insurance/assets/blockchain-a-catalyst. pdf, Erişim Tarihi*, 6:2019, 2017.

[20] Ye Liu, Yi Li, Shang-Wei Lin, and Rong Zhao. Towards automated verification of smart contract fairness. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 666–677, 2020.

[21] Zibin Zheng, Shaoan Xie, Hong-Ning Dai, Weili Chen, Xiangping Chen, Jian Weng, and Muhammad Imran. An overview on smart contracts: Challenges, advances and platforms. *Future Generation Computer Systems*, 105:475–491, 2020.

[22] Chris Dannen. *Introducing Ethereum and solidity*, volume 1. Springer, 2017.

[23] R Anandan and BS Deepak. An overview of blockchain technology: Fundamental theories and concepts. *The Convergence of Artificial Intelligence and Blockchain Technologies: Challenges and Opportunities*, pages 1–22, 2022.

[24] Analysis of the blockchain consensus algorithms. https://appinventiv.com/blog/blockchain-consensus-algorithms-guide/, Jul 2021.

[25] The merge. https://ethereum.org/en/roadmap/merge/, Apr 2023.

[26] Tasneem Darwish, Kamalrulnizam Abu Bakar, Gen Matsuda, Ahmed Aliyu, Abdul Hanan Abdullah, Abdul Samad Ismail, Ahmad Fadhil Yusof, Mohd Murtadha Mohamad, Mohd Yazid Idris, Zuhaimy Ismail, et al. Comparative analysis of blockchain consensus algorithms from shariah perspective. *Journal of Contempory Islamic Studies*, 6(1):1–21, 2020.

[27] Arati Baliga. Understanding blockchain consensus models. *Persistent*, 4(1):14, 2017.

[28] Metamask. `https://en.wikipedia.org/wiki/MetaMask`, Apr 2022.

[29] Ari Juels, Ahmed Kosba, and Elaine Shi. The ring of gyges: Investigating the future of criminal smart contracts. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 283–295, 2016.

[30] Sahil Verma and Julia Rubin. Fairness definitions explained. In *2018 IEEE/ACM International Workshop on Software Fairness (FairWare)*, pages 1–7, 2018.

[31] Shuangke Wu, Yanjiao Chen, Qian Wang, Minghui Li, Cong Wang, and Xiangyang Luo. Cream: A smart contract enabled collusion-resistant e-auction. *IEEE Transactions on Information Forensics and Security*, 14(7):1687–1701, 2018.

[32] Diceroll. `https://ropsten.etherscan.io/address/0xb95bbe8ee98a21b5ef7778ec1bb5910ea843f8f7#code`.

[33] Ethereum doubler. `https://etherscan.io/address/0x83651a62b632c261442f396ad7202fe2a4995e3a#code`.

[34] Sukrit Kalra, Seep Goel, Mohan Dhawan, and Subodh Sharma. Zeus: analyzing safety of smart contracts. In *Ndss*, pages 1–12, 2018.

[35] Anish Agnihotri Hasu. A guide to designing effective nft launches. `https://www.paradigm.xyz/2021/10/a-guide-to-designing-effective-nft-launches`, Oct 2021.

[36] A guide to designing effective nft launches. `https://bscpost.com/a-guide-to-designing-effective-nft-launches`, Oct 2021.

[37] Kentaro Sako, Shin'ichiro Matsuo, and Sachin Meier. Fairness in erc token markets: A case study of cryptokitties. In *International Conference on Financial Cryptography and Data Security*, pages 595–610. Springer, 2021.

[38] Alesja Serada, Tanja Sihvonen, and J Tuomas Harviainen. Cryptokitties and the new ludic economy: How blockchain introduces value, ownership, and scarcity in digital gaming. *Games and Culture*, 16(4):457–480, 2021.

[39] Tonya M Evans. Cryptokitties, cryptography, and copyright. *AIPLA QJ*, 47:219, 2019.

[40] Associated Press. Dapper labs, creator of nba top shot, gets $305 million in funding. https://www.espn.com/nba/story/_/id/31164860/dapper-labs-creators-nba-top-shot-get-305m-funding, Mar 2021.

[41] How do timed auctions work? – opensea help center. https://support.opensea.io/hc/en-us/articles/1500003246082-How-do-timed-auctions-work.

[42] Low fee nft marketplace - buy nfts on 5 blockchains. https://rarible.com/how-it-works/using-rarible/how-do-i-sell-my-nft-on-rarible.

[43] Mateusz Dolata, Stefan Feuerriegel, and Gerhard Schwabe. A sociotechnical view of algorithmic fairness. *Information Systems Journal*, 2021.

[44] Prabal Banerjee, Chander Govindarajan, Praveen Jayachandran, and Sushmita Ruj. Reliable, fair and decentralized marketplace for content sharing using blockchain. In *2020 IEEE International Conference on Blockchain (Blockchain)*, pages 365–370, 2020.

[45] Ken Peffers, Tuure Tuunanen, Marcus A Rothenberger, and Samir Chatterjee. A design science research methodology for information systems research. *Journal of Management Information Systems*, 24(3):45–77, 2007.

[46] Fairness definition & meaning - merriam-webster. https://www.merriam-webster.com/dictionary/fairness.

[47] Mary Jane C Samonte, Danica Grace D Advincula, Sofia Samantha B Beltran, and Aiko D Obog. Towards the development of a performance evaluation of a hyperledger fabric blockchain-based application for traceability of distribution and recall transactions in pharmaceutical supply chain.

[48] Erc1155. https://docs.openzeppelin.com/contracts/4.x/erc1155.

[49] Kerem Güle Gülen. What is the best blockchain for smart contracts and why? https://dataconomy.com/2023/03/01/what-is-the-best-blockchain-for-smart-contracts-and-why, Mar 2023.

[50] Gaurav Roy. What is avalanche (avax)?nbsp;. https://www.ledger.com/academy/topics/crypto/what-is-avalanche-avax, Jun 2023.

[51] React – a javascript library for building user interfaces. https://reactjs.org/docs/getting-started.html.

[52] Home/Installation - Page Reactstrap. https://reactstrap.github.io/.

[53] Ethereum and react cooperating. https://usedapp.io/.

[54] What is nginx? - nginx. https://www.nginx.com/resources/glossary/nginx/.

[55] What is ssl, tls and https | digicert. https://www.websecurity.digicert.com/security-topics/what-is-ssl-tls-https.

[56] Unsplash. Beautiful free images & pictures. https://unsplash.com/.

[57] Avalanche faucet. https://faucet.avax.network/.

[58] Truffle | overview - truffle suite. https://trufflesuite.com/docs/truffle/.

[59] A complete guide to ethereum's rinkeby testnet. https://www.alchemy.com/overviews/rinkeby-testnet#what-is-the-rinkeby-testnet-2.

[60] https://book.getfoundry.sh/forge/.

[61] Arthur Carvalho, Liudmila Zavolokina, Suman Bhunia, Monu Chaudhary, and Nitharsan Yoganathan. Promoting inclusiveness and fairness through nfts: The case of student-athletes and nils. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–9, 2023.

[62] Bryan White, Aniket Mahanti, and Kalpdrum Passi. Characterizing the opensea nft marketplace. In *Companion Proceedings of the Web Conference 2022*, WWW '22, page 488–496, New York, NY, USA, 2022. Association for Computing Machinery.

[63] Kailash Joshi. The measurement of fairness or equity perceptions of management information systems users. *MIS quarterly*, pages 343–358, 1989.

[64] Steven J Brams, Steven John Brams, and Alan D Taylor. *Fair Division: From cake-cutting to dispute resolution*. Cambridge University Press, 1996.

[65] Welcome to python.org. https://www.python.org/.

[66] random – generate pseudo-random numbers – python 3.10.8. https://docs.python.org/3/library/random.html.

[67] Matplotlib - visualization with python. https://matplotlib.org/.

[68] pandas - python data analysis library. https://pandas.pydata.org/.

[69] Fairness measure. https://en.wikipedia.org/wiki/Fairness_measure, Dec 2022.

Appendix A

Smart Contracts

A.1 NNM Smart Contract

```
pragma solidity >=0.4.22 <0.9.0;

import "@openzeppelin/contracts/token/ERC1155/ERC1155.sol";

contract NNM {

    // Address that deploys the smart contract, this will also be the default admin
    address constant public DEPLOYER = 0xB3493344a33B5d9dd25AdCE74e63D4D50092aDd8;

    // Variable for representing the NFT ID
    uint8 public CelebrityIDCounter = 0;

    // How many cards we want to mint for each celebrity
    uint8 private constant STARTING_QUANTITY = 3;

    // quantityRemaining specifies how many of this celebrity we can still mint - it is
        decremented with each minting of the ID
    struct Celebrity {
        uint8 id;
        uint8 quantityRemaining;
        address payable celebrityWalletAddress;
    }

    // Array of celebrities
    Celebrity[] public celebrities;

    // Tracks address of each celebrity
    mapping (uint8 => address payable) public celebrityIdToAddress;

    mapping (address => Celebrity) celebrityWalletAddressToStruct;

    // Tracks which celebrities are owned by specific accounts
    mapping (address => uint[]) ownedCardsByAccount;

    // For confirming an celebrity has been adding to the application
    event CelebrityCreated(string confirmed);

    // Function for adding to 'celebrities' array and adding more to the application
    // Can only be called from the admin feature by the smart contract deployer
    function _createCelebrity(uint8 quantity, address payable celebrityWalletAddress) public {
        // require (msg.sender == DEPLOYER);
        celebrities.push(Celebrity(CelebrityIDCounter,quantity, celebrityWalletAddress));
        celebrityWalletAddressToStruct[celebrityWalletAddress] = celebrities[celebrities.length
            -1];
        celebrityIdToAddress[CelebrityIDCounter] = celebrityWalletAddress;
        CelebrityIDCounter++;
        emit CelebrityCreated("Celebrity Successfully Created");
    }
```

```
// Admin function for removing an celebrity so that it cannot be minted again
// Doesn't remove from array, but reduces remaining quantity to 0
// Will be removed from array in buyNFT() function if we randomly get this celebrity's index
    in the array
function _removeCelebrity(address celebrityWalletAddress) public {
    // require (msg.sender == DEPLOYER);
    celebrityWalletAddressToStruct[celebrityWalletAddress].quantityRemaining = 0;
}


// Changes the quantityRemaining of the given celebrity ID
function _modifyCelebrity(uint8 idParam, uint8 newQuantity) public {
    // require (msg.sender == DEPLOYER);
    if (celebrities[idParam].id ==idParam) {
        celebrities[idParam].quantityRemaining = newQuantity;
    }
    else {
        for (uint i = 0; i < celebrities.length; i++) {
            if (celebrities[i].id == idParam) {
                celebrities[i].quantityRemaining = newQuantity;
            }
        }
    }
}


// getter
function getNumberOfCelebritys() public view returns (uint) {
    return celebrities.length;
}

// Get the ID of the celebrity at a certain array index
// These can change over time due to the architecture of the buyNFT() method
function getId(uint index) public view returns (uint) {
    return celebrities[index].id;
}

// Getter remaining quantity of celebrity at a certain array index
function getQuantityRemaining(uint index) public view returns (uint) {
    return celebrities[index].quantityRemaining;
}

// Add an Celebrity ID to our mapping 'ownedCardsByAccount'
function addOwnedCardByAccount(address adr, uint id) internal {
    ownedCardsByAccount[adr].push(id);
}

// Helper function used in the exchange feature
function removeOwnedCardByAccount(address adr, uint id) internal {
    for (uint16 i = 0; i < ownedCardsByAccount[adr].length; i++) {
        if (ownedCardsByAccount[adr][i] == id) {
            ownedCardsByAccount[adr][i] = ownedCardsByAccount[adr][ownedCardsByAccount[adr].
                length-1];
            ownedCardsByAccount[adr].pop();
            break;
        }
    }
}

// Retrieve from the 'ownedCardsByAccount' mapping given address
function getOwnedCardsByAccount(address adr) public view returns (uint[] memory) {
    return ownedCardsByAccount[adr];
}
```

```
    function getCelebrityWallet(uint index) public view returns (address payable){
        return celebrities[index].celebrityWalletAddress;
    }

    // Currently, we add 3 celebrities to the array when we deploy this smart contract - this
        will increase over time
    constructor() {
        _createCelebrity(15, payable(0x9F39bD5B715935e1957F4BAc15bBA6025b0d2F7e));
        _createCelebrity(15, payable(0x9F39bD5B715935e1957F4BAc15bBA6025b0d2F7e));
        _createCelebrity(15, payable(0x9F39bD5B715935e1957F4BAc15bBA6025b0d2F7e));
        _createCelebrity(15, payable(0x9F39bD5B715935e1957F4BAc15bBA6025b0d2F7e));
        _createCelebrity(15, payable(0x9F39bD5B715935e1957F4BAc15bBA6025b0d2F7e));
        _createCelebrity(15, payable(0x9F39bD5B715935e1957F4BAc15bBA6025b0d2F7e));
        _createCelebrity(15, payable(0x9F39bD5B715935e1957F4BAc15bBA6025b0d2F7e));
        _createCelebrity(15, payable(0x9F39bD5B715935e1957F4BAc15bBA6025b0d2F7e));
        _createCelebrity(15, payable(0x9F39bD5B715935e1957F4BAc15bBA6025b0d2F7e));
        _createCelebrity(15, payable(0x9F39bD5B715935e1957F4BAc15bBA6025b0d2F7e));
    }


}
```

A.2 CelebrityNFT Smart Contract

```
pragma solidity >=0.4.22 <0.9.0;
import "@openzeppelin/contracts/token/ERC1155/ERC1155.sol";
import "@openzeppelin/contracts/token/ERC1155/utils/ERC1155Holder.sol";
import "@openzeppelin/contracts/token/ERC1155/IERC1155.sol";
//import "@openzeppelin/contracts/token/ERC1155/extensions/ERC1155Burnable.sol";
import "./NNM.sol";

contract CelebrityNFT is ERC1155, ERC1155Holder, NNM {

    IERC1155 private _IERC1155;

    constructor()
        ERC1155(
            "https://raw.githubusercontent.com/zakatz/NFT-project-public-metadata/main/json_files
                /{id}.json"
        )
    {}

    function supportsInterface(bytes4 interfaceId) public view virtual override(ERC1155,
        ERC1155Receiver) returns (bool) {
        return super.supportsInterface(interfaceId) || interfaceId == type(IERC1155).interfaceId
            || interfaceId == type(IERC1155Receiver).interfaceId;
    }

    // BUYING SECTION OF CODE
    uint256 public randNonce = 0;
    uint256 private constant BUY_PRICE = 0.001 ether;
    uint constant rewardPercentCelebrity = 10;
    uint constant rewardPercentOwner = 5;
    // int constant percentToCelebrity = 0.10 ;
    // Payable function requires BUY_PRICE ether payment
    // Mint a **random** NFT directly to the msg.sender (buyer)
    // Add the celebrity ID to the 'ownedCardsByAccount' mapping at the msg.sender(buyer's)
        address
    // Decrement quantityRemaining of the specific Celebrity that was minted
    // If quantityRemaining has reached 0, we need to clean up our array because it is no longer
        a valid index
    // We do this by taking the Celebrity at the array.length-1 index and replacing it with the
```

```
        Celebrity at the **random** index with quantityRemaning of 0
    // Then we delete the array.length-1 entry from the array.
    function buyNFT() external payable {
        require(msg.value == BUY_PRICE);
        uint256 index = randArrayIndex(randNonce);

        // This is the second part of the removeCelebrity function which will delete an celebrity
            from the array
        // Initially, we just set their remaining quantity to 0, but now we remove that ID from
            the array if we randomly stumble accross it
        if (getQuantityRemaining(index) == 0) {
            celebrities[index] = celebrities[celebrities.length - 1];
            delete celebrities[celebrities.length - 1];

            require(getQuantityRemaining(index) != 0);
            _mint(msg.sender, celebrities[index].id, 1, "");
            addOwnedCardByAccount(msg.sender, getId(index));
            celebrities[index].quantityRemaining--;

            // Now we check again if the index is at 0
            // Once all the celebrities have been minted on a given index,
            // rearrange the celebrities array so that the last element is inserted on the index
                and
            // the length of the array is reduced by 1.
            if (celebrities[index].quantityRemaining == 0) {
                    celebrities[index] = celebrities[celebrities.length - 1];
                    delete celebrities[celebrities.length - 1];
            }
            // Now, we shll distribute the rewrd to celebrity and the rest would go
            // to the owner of the smart contrat
            uint amountToCelebrity = BUY_PRICE * rewardPercentCelebrity / 100;
            uint amountToOwner = BUY_PRICE - amountToCelebrity;
            getCelebrityWallet(index).transfer(amountToCelebrity);
            payable(DEPLOYER).transfer(amountToOwner);
        }
            // We expect this else to trigger more often
            else {
            require(getQuantityRemaining(index) != 0);
            _mint(msg.sender, celebrities[index].id, 1, "");
            addOwnedCardByAccount(msg.sender, getId(index));
            celebrities[index].quantityRemaining--;

            // Once all the celebrities have been minted on a given index,
            // rearrange the celebrities array so that the last element is inserted on the index
                and
            // the length of the array is reduced by 1.
            if (celebrities[index].quantityRemaining == 0) {
                    celebrities[index] = celebrities[celebrities.length - 1];
                    delete celebrities[celebrities.length - 1];
            }
            // Now, we shll distribute the rewrd to celebrity and the rest would go
            // to the owner of the smart contrat
            uint amountToCelebrity = BUY_PRICE * rewardPercentCelebrity / 100;
            uint amountToOwner = BUY_PRICE - amountToCelebrity;
            getCelebrityWallet(index).transfer(amountToCelebrity);
            payable(DEPLOYER).transfer(amountToOwner);
        }
    }

    // Generate a 0 <= random number < array.length
    function randArrayIndex(uint256 _modulus) internal returns (uint256) {
        uint256 randomNumber = uint256(
            keccak256(abi.encodePacked(msg.sender, _modulus))
        );
        randNonce++;
```

```
        return randomNumber % celebrities.length;
    }


    // EXCHANGE SECTION OF CODE
    uint32 constant MAXVAL_32 = 4294967295;
    uint32 exchangeCounter = 0;
    uint32 TIMEOUT = 86400;
    struct ExchangeInfo {
        address payable buyer;
        address payable seller;
        int buyerCelebrityId;
        uint sellerCelebrityId;
        uint price;
        uint expiryTime;
    }
    mapping(uint32 => ExchangeInfo) hashToInfo;
    // temporary
    uint[] hashArray;

    event TxCount(uint32 counter);

    //Exchanges are done on an escrow system with a secret hash
    //The person who intializes the exchange will never be spending any ETH, they are the seller
    //The seller transfers their Celebrity to the smart contract to hold as escrow
    function initializeExchange(address buyer, uint sellerCelebrityId, int buyerCelebrityId, uint
         price) external returns(uint) {
        setApprovalForAll(buyer, true);
        setApprovalForAll(address(this), true);
        hashToInfo[exchangeCounter%MAXVAL_32] = ExchangeInfo(payable(buyer), payable(msg.sender),
             buyerCelebrityId, sellerCelebrityId, price, block.timestamp + TIMEOUT);
        emit TxCount(exchangeCounter%MAXVAL_32);
        hashArray.push(exchangeCounter);
        exchangeCounter++;
        safeTransferFrom(msg.sender, address(this), sellerCelebrityId, 1, "");
        removeOwnedCardByAccount(msg.sender, sellerCelebrityId);
        return exchangeCounter%MAXVAL_32;
    }

    //The buyer finalizes the exchange - there are 3 options
    // 1. Buyer gets Seller's Celebrity in exchange for ETH
    // 2. Buyer gets Seller's Celebrity in exchange for their own Celebrity (we can still call
        them the buyer)
    // 3. Buyer gets Seller's Celebrity in exchange for ETH & their own Celebrity
    // The transaction details, ExchangeInfo info, are accessible via the secret hash
    // If the Buyer does not have the secret hash and the correct exchange information, the
        transfer will not go through
    function finalizeExchange(address seller, uint sellerCelebrityId, int buyerCelebrityId, uint
        price, uint32 exchangeVal) external payable {
        ExchangeInfo memory info = hashToInfo[exchangeVal%MAXVAL_32];
        require (payable(seller) == info.seller);
        require (payable(msg.sender) == info.buyer);
        require (sellerCelebrityId == info.sellerCelebrityId);
        require (block.timestamp < info.expiryTime);
        require (buyerCelebrityId == info.buyerCelebrityId);
        require (price >= info.price && msg.value == price);

        setApprovalForAll(seller, true);
        setApprovalForAll(address(this), true);

        // price = price / 1000 ether;
        if (price > 0) {
            //transfer money to celebrity
            uint rewardAmountToCelebrity = price * rewardPercentCelebrity / 100;
            // Transfer reward money to owner
```

```
            uint rewardAmountToOwner = price * rewardPercentOwner / 100;
            // Transfer remaining reward to seller
            uint amountToSeller = price - rewardAmountToCelebrity - rewardAmountToOwner;
            if (buyerCelebrityId < 0) {
                celebrityIdToAddress[uint8(sellerCelebrityId)].transfer(rewardAmountToCelebrity);
            } else {
                celebrityIdToAddress[uint8(sellerCelebrityId)].transfer(rewardAmountToCelebrity /
                     2);
                celebrityIdToAddress[uint8(uint256(buyerCelebrityId))].transfer(
                    rewardAmountToCelebrity / 2);
            }
            payable(DEPLOYER).transfer(rewardAmountToOwner);
            info.seller.transfer(amountToSeller);
        }

        // the buyer is only paying money
        if (buyerCelebrityId < 0) {
            //safeTransferFrom(address(this), msg.sender, sellerCelebrityId, 1, "");
            _mint(msg.sender, sellerCelebrityId, 1, "");
            addOwnedCardByAccount(msg.sender, sellerCelebrityId);
        }
        // the buyer is only exchanging card. No money involved
        else if (info.price == 0 && buyerCelebrityId >= 0) {
            uint newBuyerCelebrityId = uint(buyerCelebrityId);
            _mint(msg.sender, sellerCelebrityId, 1, "");
            safeTransferFrom(msg.sender, seller, newBuyerCelebrityId, 1, "");
            removeOwnedCardByAccount(msg.sender, newBuyerCelebrityId);
            addOwnedCardByAccount(seller, newBuyerCelebrityId);
            addOwnedCardByAccount(msg.sender, sellerCelebrityId);
        }
        // the buyer is exchanging with card and money
        else {
            uint newBuyerCelebrityId = uint(buyerCelebrityId);
            //safeTransferFrom(address(this), msg.sender, sellerCelebrityId, 1, "");
            _mint(msg.sender, sellerCelebrityId, 1, "");
            safeTransferFrom(msg.sender, seller, newBuyerCelebrityId, 1, "");
            removeOwnedCardByAccount(msg.sender, newBuyerCelebrityId);
            addOwnedCardByAccount(seller, newBuyerCelebrityId);
            addOwnedCardByAccount(msg.sender, sellerCelebrityId);
        }
        delete hashToInfo[exchangeVal%MAXVAL_32];
    }

    // Shows the celebrity ID that your friend has posted to exchange
    // Use this on the front end to finalize the exchange with the proper celebrities
    function getFriendPostedCelebrity(uint32 exchangeVal) public view returns (uint) {
        return hashToInfo[exchangeVal%MAXVAL_32].sellerCelebrityId;
    }

    // Method to get the exchange info from the hash value
    function getFriendAddress(uint32 exchangeVal) public view returns (address) {
        ExchangeInfo memory info = hashToInfo[exchangeVal%MAXVAL_32];
        return info.seller;
    }

    // Method to get the exchange info from the hash value
    function getCelebrityYouGive(uint32 exchangeVal) public view returns (int) {
        ExchangeInfo memory info = hashToInfo[exchangeVal%MAXVAL_32];
        return info.buyerCelebrityId;
    }

    // Method to get the exchange info from the hash value
    function getPriceYouPay(uint32 exchangeVal) public view returns (uint) {
        ExchangeInfo memory info = hashToInfo[exchangeVal%MAXVAL_32];
        return info.price;
```

```
    }

    // Method to get the exchange info from the hash value
    function getExchangeInfo(uint32 exchangeVal) public view returns (address, address, int, uint
        , uint, bool) {
        ExchangeInfo memory info = hashToInfo[exchangeVal%MAXVAL_32];
        bool isExpired = false;
        if (info.expiryTime < block.timestamp) isExpired = true;
        return (info.seller, info.buyer, info.buyerCelebrityId, info.sellerCelebrityId, info.
            price, isExpired);
    }

    // Method to get all the hash values
    function gethashArray() public view returns (uint[] memory) {
        return hashArray;
    }

    // To undo exchanges that have been initialized
    function recallCelebrity(uint32 exchangeVal) public {
        ExchangeInfo memory info = hashToInfo[exchangeVal%MAXVAL_32];
        // require (block.timestamp > info.expiryTime);
        require (msg.sender == info.seller);
        _mint(msg.sender, info.sellerCelebrityId, 1, "");
        addOwnedCardByAccount(msg.sender, info.sellerCelebrityId);
        delete hashToInfo[exchangeVal%MAXVAL_32];
    }

}
```

A.3 CelebrityNFTAuction Smart Contract

```
pragma solidity >=0.4.22 <0.9.0;
import "./CelebrityNFT.sol";

contract CelebrityNFTAuction is CelebrityNFT {

     uint32 bidCounter = 0;

    event BidCounter(uint bidCounter);

    struct AuctionInfo {
        address payable seller;
        address payable highestBidder; // should be payable?
        uint celebrityId;
        uint maxBidAmount;
        // uint minBidAmount;
        uint highestBidAmount;
        uint bidExpiryTime;
        bool isOpen;
    }
    mapping(uint32 => AuctionInfo) auctionIdToInfo;
    uint[] auctionArray;

    event TransferSuccess(bytes4 returnval);

    // Auction are done on an escrow system with a secret hash
    // The person who intializes the auction will never be spending any ETH, they are the seller
    // The seller transfers their Celebrity to the smart contract to hold as escrow
    function initializeAuction(uint celebrityId, uint maxBidAmount, uint minBidAmount, uint
        timeout) external returns(uint) {
        require(maxBidAmount > minBidAmount);
        setApprovalForAll(address(this), true);
```

```
        // make sure the highestbidamount is not negative
        auctionIdToInfo[bidCounter%MAXVAL_32] = AuctionInfo(payable(msg.sender), payable(address
            (0)),
                celebrityId, maxBidAmount, minBidAmount - 1, block.timestamp + timeout, true);

        emit TxCount(bidCounter%MAXVAL_32);
        auctionArray.push(bidCounter%MAXVAL_32); // bidCounter%MAXVAL_32
        emit BidCounter(bidCounter);
        bidCounter++;

        safeTransferFrom(msg.sender, address(this), celebrityId, 1, "");
        emit TransferSuccess(onERC1155Received(msg.sender, msg.sender, celebrityId, 1, ""));
        removeOwnedCardByAccount(msg.sender, celebrityId);

        emit BidCounter(bidCounter);

        return bidCounter%MAXVAL_32;
    }

    // Method to get all the open bids
    function getOpenBids() public view returns (uint[] memory) {
        return auctionArray;
    }

    function getAuctionInfoByID(uint32 auctionID) public view returns (address, address, uint,
        uint, uint, bool, bool) {
        AuctionInfo memory info = auctionIdToInfo[auctionID%MAXVAL_32];
        bool isExpired = false;
        if (info.bidExpiryTime < block.timestamp) isExpired = true;
        return (info.seller, info.highestBidder, info.celebrityId, info.maxBidAmount, info.
            highestBidAmount, info.isOpen, isExpired);
    }

    event HighestBidder(address highestBidder);

    function bid(uint32 auctionID) external payable {
        AuctionInfo memory info = auctionIdToInfo[auctionID%MAXVAL_32];
        emit HighestBidder(info.highestBidder);

        require(info.isOpen == true);
        require (block.timestamp < info.bidExpiryTime);
        require (msg.value > info.highestBidAmount);

        if (info.highestBidder != address(0)) {
            info.highestBidder.transfer(info.highestBidAmount);
        }

        auctionIdToInfo[auctionID%MAXVAL_32].highestBidder = payable(msg.sender);
        auctionIdToInfo[auctionID%MAXVAL_32].highestBidAmount = msg.value;

        emit BidCounter(msg.value);
        emit BidCounter(info.maxBidAmount);
        // works well upto here

        // when ....
        if (msg.value >= info.maxBidAmount) {
        emit HighestBidder(info.highestBidder);

            claimNFTByBidder(auctionID, msg.sender);
        }

    }

    function claimNFTByBidder(uint32 auctionID, address winner) public {
        AuctionInfo memory info = auctionIdToInfo[auctionID%MAXVAL_32];
```

```
        emit HighestBidder(info.highestBidder);
        emit HighestBidder(winner);

        require(info.isOpen == true);
        require(info.highestBidder == payable(msg.sender) || info.highestBidder == winner);
        require(block.timestamp > info.bidExpiryTime || info.highestBidAmount >= info.
            maxBidAmount);
        emit BidCounter(info.highestBidAmount);

        auctionIdToInfo[auctionID%MAXVAL_32].isOpen = false;

        setApprovalForAll(info.seller, true);
        // setApprovalForAll(msg.sender, true);

        emit HighestBidder(address(this));

        // address(this) is the contract address.
        // setApprovalForAll(address(this), true);          // ERROR

        uint price = info.highestBidAmount;
        //transfer money to celebrity
        uint rewardAmountToCelebrity = price * rewardPercentCelebrity / 100;
        // Transfer reward money to owner
        uint rewardAmountToOwner = price * rewardPercentOwner / 100;
        // Transfer remaining reward to seller
        uint amountToSeller = price - rewardAmountToCelebrity - rewardAmountToOwner;
        //transfer money
        celebrityIdToAddress[uint8(info.celebrityId)].transfer(rewardAmountToCelebrity);
        // payable(DEPLOYER).transfer(rewardAmountToOwner);
        info.seller.transfer(amountToSeller);

        _mint(info.highestBidder, info.celebrityId, 1, "");
        addOwnedCardByAccount(info.highestBidder, info.celebrityId);

        delete auctionIdToInfo[auctionID%MAXVAL_32];
    }

    function revokeAuction(uint32 auctionID) external {
        AuctionInfo memory info = auctionIdToInfo[auctionID%MAXVAL_32];

        emit BidCounter(info.bidExpiryTime);
        emit BidCounter(block.timestamp);

        require(info.bidExpiryTime < block.timestamp);
        require(info.isOpen == true);
        require(info.seller == msg.sender);
        require(info.highestBidder == payable(address(0)));

        auctionIdToInfo[auctionID%MAXVAL_32].isOpen = false;

        _mint(msg.sender, info.celebrityId, 1, "");
        addOwnedCardByAccount(msg.sender, info.celebrityId);
        delete auctionIdToInfo[auctionID%MAXVAL_32];
    }

}
```