BLOCKCHAIN APPLICATION ON LOYALTY CARD

A Thesis Submitted to
the Graduate School of Engineering and Sciences of
İzmir Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

in Computer Engineering

by Osman SÖNMEZTÜRK

April 2020
İZMİR

We approve the thesis of **Osman SÖNMEZTÜRK**

Examining Committee Members:

Assoc. Prof. Tolga AYAV
Department of Computer Engineering, Izmir Institute of Technology

Assist. Prof. Serap ŞAHİN
Department of Computer Engineering, Izmir Institute of Technology

Assoc. Prof. Ahmet Tuncay ERCAN
Department of Computer Engineering, Yaşar University

8 April 2020

Assoc. Prof. Tolga AYAV
Supervisor, Department of Computer Engineering/Izmir Institute of Technology

Prof. Yusuf Murat ERTEN
Co-Supervisor, Department of Computer Engineering/Bakırçay University

Assoc. Prof. Tolga AYAV
Head of the Department of Computer Engineering/Izmir Institute of Technology

Prof. Mehtap EANES
Dean of the Graduate School of Engineering and Sciences

ACKNOWLEDGMENTS

I would like to express my gratitude to my advisor Prof. Yusuf Murat ERTEN for his continuous support and effort during my graduate studies. He has always mentored me with all his sincerity and kindness. I would also like to thank Assoc. Prof. Tolga AYAV for his support and supervision during my thesis work.

I am very thankful for Begüm ÜNSAL and my family, who are always behind me, for being supportive during this period.

I also would like to thank Izmir Institute of Technology for facilitating my research by providing open library resources.

ABSTRACT

BLOCKCHAIN APPLICATION ON LOYALTY CARD

Today, traditional loyalty systems are insufficient to meet the needs of users. The users need to stay within the loyalty system for a long time and accumulate points in order to win prizes and besides, the rewards they receive may be out of their interest. In addition, users usually forget the awards they have won in traditional loyalty systems and have difficulty in following up rewards. In addition to that, users usually do not prefer to share their personal information to join loyalty systems due to privacy concerns. Therefore, the number of customers in the loyalty systems is decreasing day by day. The designed loyalty program mentioned in this thesis works with IZTECH Tokens, which works on the Ethereum chain and are created by following ERC20 standards. Thanks to the new generation loyalty system, users can convert their earned tokens to Ethereum on the stock exchange without accumulating them or can receive services or products with the accumulated tokens according to their interests from a market that has been contracted by the manufacturer. Additionally, users in the designed system do not need to carry many cards, it is adequate to have only one Ethereum wallet. Furthermore, users do not need to share any personal data to join the loyalty system. Markets can request Ether from the manufacturer according to the number of tokens they have accumulated. The loyalty system mentioned in this thesis not only aims to establish a win-win relationship between the manufacturer, market, and client but also to find solutions to the customer problems mentioned above.

ÖZET

SADAKAT KARTI ÜZERİNDE BLOK ZİNCİRİ UYGULAMASI

Günümüzde gelenekselleşmiş sadakat zinciri uygulamaları sistemin içerisinde yer alan kullanıcıların isteklerini karşılamamaya başlamıştır. Kullanıcıların sadakat programlarında ödül kazanabilmek için yüksek miktarda puan biriktirmeleri gerekmektedir ve çoğunlukla kazanılan ödüller kullanıcıların ilgi alanları dışında olmaktadır. Ayrıca, kullanıcılar geleneksel sadakat sistemlerinde kazanmış oldukları ödülleri de unutmakta ve takip etmekte zorlanmaktadırlar. Günümüzde kullanıcılar gizlilik konusundaki tereddütlerden dolayı sadakat sistemlerine katılırken bilgilerini paylaşmamayı tercih etmektedirler. Bu yüzden sadakat programları gün geçtikçe hem kullanıcıların hem de büyük şirketlerin güvenini kaybetmektedir. Bu tezde bahsedilen yeni nesil sadakat programı, Ethereum zinciri üzerine kurulmuş olup ve ERC20 standartları takip edilerek oluşturmuş IZTECH jetonları ile çalışmaktadır. Oluşturulan akıllı sözleşme sayesinde sistem içerisindeki jetonlar kontrol edilebilir ve kullanıcılar jetonlarını bahsedilen sözleşme üzerinden transfer edebilir. Sistemin içerisindeki kullanıcılar biriktirdikleri jetonları borsada Ethereum'a dönüştürebilmekte veya anlaşmalı olan marketlerden biriktirilen jetonlar ile ürün veya hizmetleri ilgi alanlarına göre satın alabilmektedirler. Ayrıca dizayn edilen sistemin içerisinde bulunan kullanıcıların yanlarında birden fazla kart taşımalarına gerek yoktur, sadece bir tane Ethereum cüzdanlarının olması yeterlidir. Buna ek olarak kullanıcıların bahsedilen sadakat sistemine katılabilmesi için kişisel verilerini paylaşmalarına da gerek yoktur. Marketler ise biriktirdikleri jeton miktarına bağlı olarak üretici firmadan Ether alabilmektedirler. Makalede bahsedilen yeni nesil sadakat programı, üretici firma, market ve kullanıcı arasında kazan kazan ilişkisi kurmayı hedeflemiştir ve yukarda bahsedilen müşteri sorunlarına çözüm bulmayı hedeflemiştir.

TABLE OF CONTENTS

LIST OF FIGURES

Figure **Page**

LIST OF TABLES

Table **Page**

LIST OF ABBREVIATIONS

API ..... Application Programming Interface
BCOIP ..... Blockchain-Based Cross-Organizational Integrated Platform
dApps ..... Decentralized Applications
dPoS ..... Delegated Proof of Stake
EOS ..... Ethereum Operating System
ERC ..... Ethereum Request for Comments
EVM ..... Ethereum Virtual Machine
ID ..... Identification
IoT ..... Internet of Things
npm ..... Node Package Manager
prc ..... Peer-Review Coin
pBFT ..... Practical Byzantine Fault Tolerance
PoS ..... Proof of Stake
PoW ..... Proof of Work
TxID ..... Transaction ID
QR ..... Quick Response

CHAPTER 1

INTRODUCTION

Nowadays, people have started to look for new alternatives to reduce dependence on banks, and the Internet is the most obvious medium to be used. However, the community could not trust the internet, especially when it comes to payment transactions although it is very easy and almost free to exchange anything on the internet. This led to a search for an alternative and reliable currency which could not be copied or replicated in the virtual world. Thus, the concept of Bitcoin was first introduced. This concept, in which blocks are connected to each other with one-way functions, was introduced in 2009 by an entity known as Satoshi Nakamoto.

Although the idea of Blockchain was originally proposed as a payment system, it quickly became popular due to its difficult to break and easily maintained structure. As a result, the Blockchain system enabled the use of cryptocurrencies on the Internet. Blockchain technology is not only seen as a cryptocurrency but it has also been used in areas such as IoT, security, and voting system [1,2].

With the increasing use of the internet, electronic voting systems started moving to the Internet. In addition, electronic voting systems began to work on a non-trusting Blockchain network [3].

Loyalty system is a system that was established to encourage the customers of the manufacturers to buy products or services. Loyalty system usage also became more popular recently. According to the recent research, the number of members in the loyalty system reached 3.8 billion in 2016. 57% of the members think that it takes a long time to win a reward in the loyalty programs, and 53% think that the awards given in the loyalty systems are beyond their interests [4].

Blockchain has started to be used to meet the loyalty system's new expectations and make it reliable. With the development of Blockchain technology, the loyalty system allowed exchange between not only one manufacturer but several manufacturers, and a loyalty system was established on the Blockchain network to meet business to business needs.

On the Blockchain, transactions can be made not only with currencies but also with tokens. In addition, expiration data can be determined for a token or different ID's can be given to tokens based on their usage area. The loyalty systems use tokens instead of cryptocurrencies because not only more resources such as time, money and energy are consumed to generate all the cryptocurrencies. Because tokens act like an object on the Ethereum chain. In other words, many tokens as desired can be created on Solidity. On the other hand, mining is required in order to produce a certain amount of cryptocurrencies. Because of these features, they are more useful for loyalty systems than cryptocurrencies.

Using tokens, the manufacturer can define a specific market strategy. Producers can use these tokens to attract their customers into their loyalty systems, and it is aimed to achieve the intended win-win relationship. Not only the customers in the system do not have to carry a lot of cards with them physically, but also there are no restrictions on rewards, as in the current loyalty systems. The creation fee of the tokens is close to zero and there only exists a fee to run the function on the Ethereum chain. Although the manufacturer is the one who decides and determines the market strategy, the high number of tokens in the market will decrease the value of the token.

In order to provide the use of tokens by users, and by considering security issues, the standards working on the Ethereum network were followed. In general, users can access to Rinkeby network, which is the test network that is accessible to anyone in the Blockchain technology and run the smart contract functions within the loyalty system through the web interface. When customers within the established loyalty system realize that they earn as much as they spend, they tend to spend more. Thus, both the market and the manufacturer make profit. Additionally, the permanence of the customers within the system is provided.

1.1. Motivation

While technology is developing fast, the systems that were established in the past had to keep up with the new technologies. Traditional loyalty systems are not enough to meet the demands of customers, moreover, in a world where everything is renewed,

traditional loyalty systems cannot excite customers who do not want to stay in the system. As a result of the survey where 177 people participated in a different study, 86% found the idea of winning ether in a loyalty system based on Blockchain technology attractive and 67% stated that they wanted to keep their money in the ether [5]. The article mentioned the new generation loyalty system, which is adapted using Blockchain technology. Although many companies use loyalty chains working on Blockchain, they do not explain their smart contracts, or the structures used due to commercial concerns. In this article, research has been done on how new loyalty systems can be used. The system proposed aims to offer a new alternative to the customers, markets and, manufacturers.

As can be seen in the literature review chapter, there are many studies that have aimed to integrate loyalty systems into Blockchain. Overall, the results from these studies show that customers are satisfied with the new generation loyalty systems. Our motivation to develop the loyalty system proposed in the article is to ensure the satisfaction of the customers as well as the satisfaction of the manufacturer and the market in the system.

Although 3.8 billion loyalty users were reached in 2017, research studies show that loyalty program growth is less than in the past. Given the 3.8 billion loyalty program membership, consumers' most important problem with loyalty programs is that it takes a long time to earn points [4]. It is foreseen that the loyalty program based on Blockchain and cryptocurrency will solve this important problem by instantly uploading awards to the users' accounts.

In this thesis, it is aimed to give information on the research studies on the new generation loyalty system. The loyalty tokens produced were derived from the ERC20 standards and were written using a smart contract with the latest Solidity version [6]. Ethereum chain was preferred for the system, due to its large developer community and fast block production. In addition, a proposal for the win-win relationship between the market, the customer and the manufacturer has been presented within the established loyalty system.

1.2. Outline of the Thesis

The thesis is organized as follows: Chapter 2 contains information about Blockchain, a decentralized application created using Blockchain, smart contracts specifically for Ethereum chain and general information about the loyalty system. Chapter 3 provides a general literature overview. This part contains the general studies that have a similar market strategy or use similar technologies. The proceeding chapter contains how the installed system works and information about the technology and development used in the loyalty system. Chapter 5 contains information about the differences between the current research studies and the newly developed system, provides final remarks and gives information about future research.

CHAPTER 2

BACKGROUND

2.1. Blockchain

Blockchain is a popular technology nowadays and there are many areas where it is applied. There are Blockchain technologies customized according to their fields, and the most frequently used ones are Bitcoin, Ethereum and Tron. Blockchain is started to be used in many fields, especially in the field of finance [7].

Until the 19th century, there were unilateral contracts in which the owner of the contract was the ruler of a country and had the whole control as the owner of the contract. For example, the king could change and apply the contract clauses as he wanted because there was no system controlling it. Later, under the supervision of joint authority, the concept of the bank was introduced, in which the two parties signed contracts and the bank supervised the rules. However, this structure has started to lose its validity recently with the decrease in trust in the banks. Thus, a structure without a central authority started to be accepted. Blockchain is basically a network where transactions are registered to decentralized blocks and blocks are connected to each other with cryptographic hash functions, and every user can see the process and be involved. Without a 3rd party in the Bitcoin payment system, people can pay each other, and these transactions are visible to everyone within the established network. This network can simply be defined as a chain formed by connecting blocks of transaction information. Each block takes the hash of the previous block and puts it into the hash function and transfers it to the other block, making it difficult to break the chain [8].

Firstly, Satoshi Nakamoto's Blockchain technology is more reliable than many currencies created on the internet because it is the only technology that solves the double-spending problem. Double spending is called a currency expendable twice and it is very easy to copy only strings from within the chain network [9]. Blockchain offered a solution to the double-spending problem [10].

In the Blockchain system, there is a protocol between the sender and the receiver based on the asymmetric cryptography algorithm and this is called a digital signature. This protocol guarantees to the buyer that the information sent by the sender has not changed and that it has also come for the buyer. The basis of the unbreakable asymmetric cryptography is that the elliptic curve discrete logarithm problems are very difficult to solve [11]. The sender generates two major prime numbers within its own rules, and these become private and public keys of the sender. If any plain text is encrypted with one of these keys, only the key which is its pair can convert the ciphertext to plain text [12]. Thanks to this, the recipient makes sure that the message is not seen by the 3rd party.

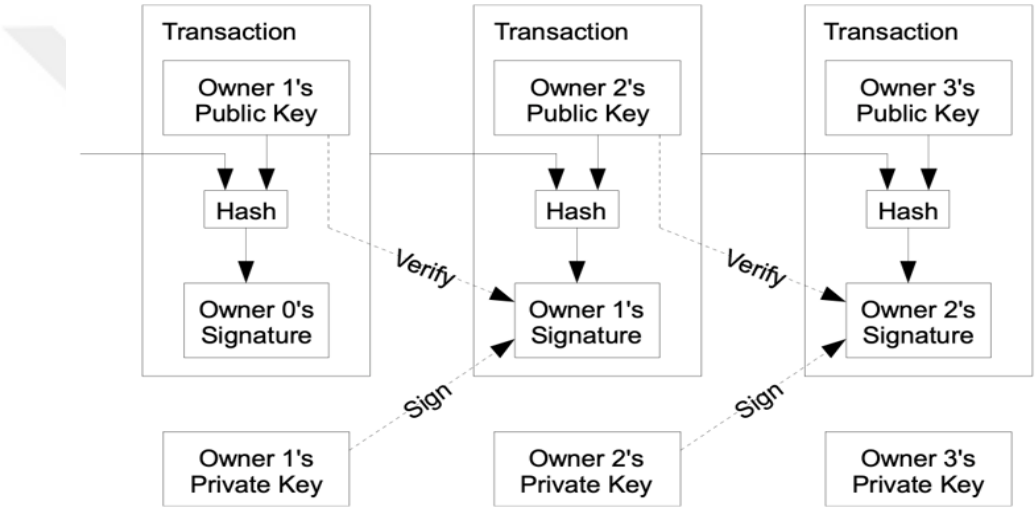

Figure 2.1. Digital Signature [10]

When the coin owner transfers money to another user in the Blockchain system, transaction information is put into the hash function and encrypted with the public key of the receiver and sent to the recipient with transaction information in plain text. Using its private key, the receiver can decode the encrypted message and see its hashed version. In addition to the encrypted message, the transaction information sent takes the hash with the hash function and compares the two hash outputs. If the two hash outputs are equal, one can be sure that the transaction information is correct and has not changed [13].

This hash result is then added to the end of the coin. Therefore, every coin is like a notary, as if it contains a record of all previous owners of a car. This process alone is not enough to solve double spending. In addition to the cryptographic hash, each

transaction is announced to the miners on the network and enters the unapproved transaction pool. Miners make proof of work or proof of stake according to consensus algorithm and write transactions into blocks. If more than one block is generated simultaneously and two parallel chains are created, then the chain which is longest is considered as the correct chain and transactions are verified [14].

Hash functions are very important for Blockchain. The hash functions are the reason why the Blockchain is unbreakable and its operations cannot be changed. Hashing is the processing of data in a block through a mathematical function and as a result, produces a fixed-length output. The hash functions are one-way functions, which means it is mathematically impossible to obtain input using output and producing a fixed-length output increases security because anyone trying to decrypt the hash cannot tell how long the input is by looking at the length of the output [15].

The block header contains the following information: hash of the block header, the hash of all transactions recorded in the block, the hash of the parent block, date and time of block creation, parameter bits and nonce, and block version. The header also includes the transaction hash of the current block. It is calculated by using an algorithm, which is known as the Merkle tree (binary hash tree). The Merkle tree illustration can be seen below in the Figure 2.2.

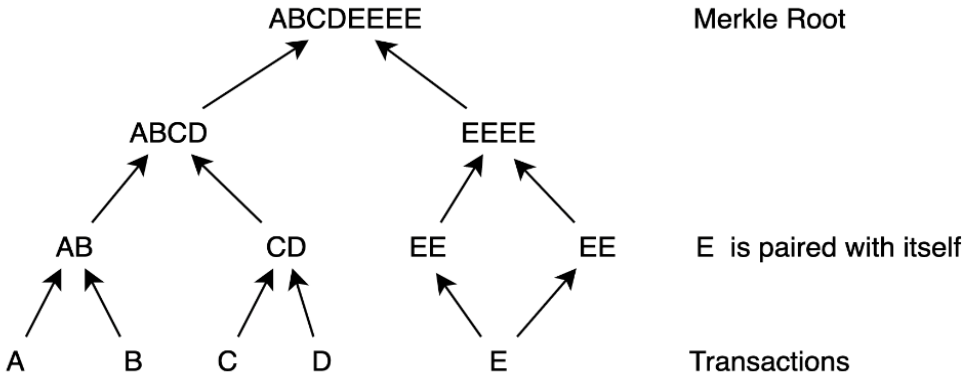

Figure 2.2. Merkle Tree

The output of transactions from the hash function is called TxID, and each TxID is paired with another TxID and then combined to create a Merkle tree. If there is an odd number of TxID, TxID without a partner is combined with a copy of itself. The merging process continues until only one TxID remain [16].

Thus, transaction information is guaranteed to be unchanged in the previous blocks and a great memory gain is provided. For example, in order to verify that transaction D has been added to the block, in addition to root, only a copy of C, AB and EEEE hashes are needed so, the client does not need to know any of the other transactions. If all five operations in this block are at maximum size, downloading the entire block requires more than 500,000 bytes, but downloading the block header as well as three hashes only requires 140 bytes.

There are mechanisms that ensure the security of the coins used on the Blockchain and prevent malicious users. These mechanisms are called consensus mechanisms and they vary according to the types of coins. Some of these can be listed as Proof of Work (PoW), Proof of Stake (PoS), practical Byzantine Fault Tolerance (pBFT), and delegated Proof of Stake (dPoS) [14].

In the Proof of Work consensus mechanism, each miner generates hash using the Version, Previous Block Hash, Merkle Root, Timestamp, Difficulty Bits, and Nonce values and changes the nonce so that the generated value is within certain rules. Changing one bit in the values entering the hash function causes the result to be completely different. Therefore, miners increase the nonce value one by one and try to produce the output within the rules. The miner, who finds the appropriate number, wins certain percentages and announces to the Network that the block is created. Other miners, after verifying the generated block, start calculating the hash again to create the next block using the hash of the newly found block, and so on. It takes about 10 minutes to find the correct hash and it is recommended to build at least 6 blocks to ensure the accuracy of a transaction of the pow. This slowness in block production may cause slow operation of the system. In addition, each block created increases the difficulty of generating the next block [17].

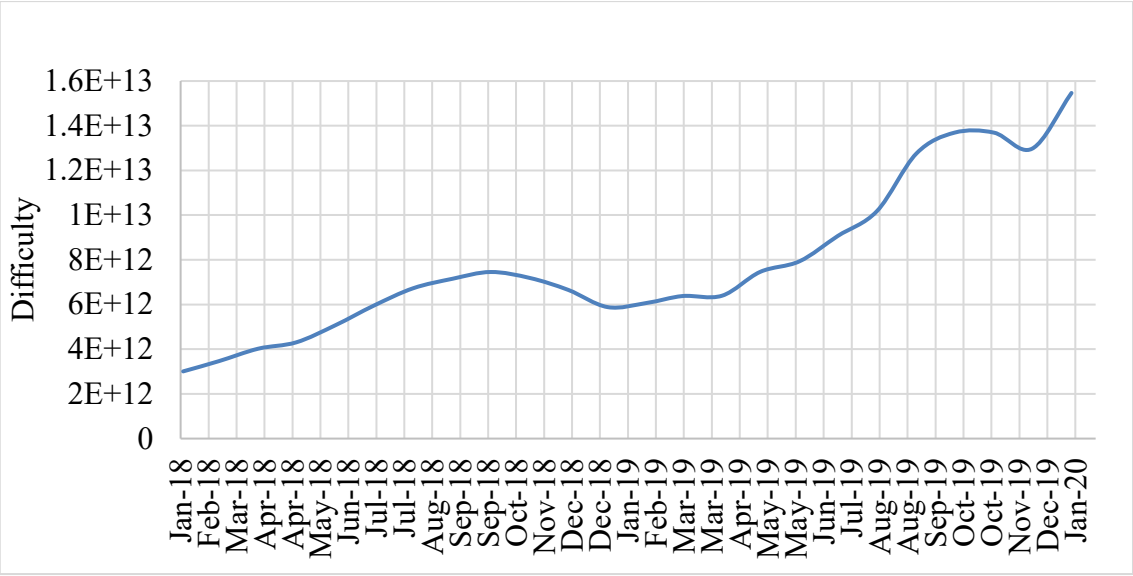

Figure 2.3. Block Creation Difficulty Change

In the Proof of Stake consensus mechanism, the verification process is done randomly depending on the number of coins in the person's wallet account. So, every wallet with the coin is also a verifier. Of course, accounts with many coins in the system are more likely to be used as confirmatory and therefore receive more shares from each transaction. It takes 15 seconds to produce the correct block with this mechanism. Ethereum uses this mechanism as consensus detection [18].

The Practical Byzantine Fault Tolerance consensus mechanism has found a solution to the Dilemma of Byzantine Generals Problem. This dilemma was designed in 1982 and assumes that each general has its own army and that each group is deployed at different points in the city they plan to attack. The generals must agree on either attacking or withdrawing. It is insignificant to attack or withdraw as long as an agreement is reached, the important thing is to implement the joint decision in a coordinated manner.

The algorithm developed for the solution of this problem is adapted as the Blockchain consensus algorithm. In the Practical Byzantine Fault Tolerance consensus algorithm, the majority of nodes in a distributed network must agree on the same action and implement it. Nodes in a pBFT enabled distributed system is sequentially aligned so that a node is named primary and the others are secondary. The aim is to help all honest nodes reach a consensus about the state of the system using the majority rule. As the number of nodes in the system increases, the system becomes secure and at least 2-3 votes are required for the system to approve [19].

Delegated Proof of Stake consensus mechanism is based on a voting system where shareholders transfer their work to third parties. In other words, these shareholders have the right to vote for several delegates who will protect the system on their behalf. Delegates are responsible for providing consensus during the creation and verification of new blocks. The voting power of the user is directly proportional to the number of coins they have. Generally, the awards received by the delegates are shared in proportion with the people who have chosen them. Since the dPoS system is maintained by voters, delegates tend to be honest and effective, otherwise, they are excluded [20].

The smart contract layer is responsible for compiling, distributing, implementing, executing conditional triggering and automatic execution of specified rules, by coding the business logic of the Blockchain system to minimize manual intervention. It is difficult to change the data of the smart contract after it is migrated to the Blockchain. Smart Contracts are divided into two main groups as Turing complete and Turing incomplete. Smart contracts written as Turing complete are programmable and feasible for complex business logic. On the other hand, smart contracts written with the Turing incomplete approach are simpler, efficient and safe. In order to code the complex market strategy into a smart contract in the designed loyalty system, the solidity written as Turing complete was preferred. Some smart contract languages and coin types are as follows; Bitcoin, Ethereum, EOS, and Hyperledger are programmed with Bitcoin Script, Solidity, C++, and Fabric Go, respectively. More explanation will be provided on smart contracts in the proceeding section.

Access control groups change according to the goals of Blockchains. The first is the decentralized chain, which is public. There are many unknown users who have the right to write and read in such chains. The energy consumption of this chain is high, slow but reliable. On the other hand, users of private chains are known, but the authorization to write is only in the central authority. Although such chains are fast and stable, their safety is less than that of public chains and they have a more centralized structure [21].

Due to the increase in the application of Blockchain day by day, it has a very rapidly growing community. Important factors for decentralized applications include security, transaction per second, and stability, so consensus detection improves as decentralized applications grow and new coin varieties are emerging. At the same time, by the end of January 2020, the number of decentralized applications (dApps) has increased to 3,315 and goes up every day [22]. The table below demonstrates the number of daily active users and transactions by platforms.

Table 2.1. The Number of Daily Active Users and Transactions by Platforms [22]

Platform	Total dApps	Daily Active Users	Transactions (24h)	Volume (24h)	Number of Contracts
Ethereum	2,760	19.47k	81.51k	34.58k	4.24k
EOS	321	0	0	0	498
Steem	94	11.29k	424.79k	4679k	166
Klaytn	44	55.08k	418.11k	0	106
Blockstack	21	Unknown	Unknown	Unknown	0
NEO	20	Unknown	Unknown	Unknown	31
POA	19	178	3.07k	0	48
Loom	15	Unknown	Unknown	Unknown	75
xDai	12	6	30	12	39
GoChain	7	1	2	0	17
OST	2	46	250	9.8k	2

Today, the first introduced bitcoin reached around $ 171 billion market cap, which ranks first according to the market cap amount. Ethereum is in second place, with a market cap of nearly $ 20 billion [23].

2.2. Smart Contract

The smart contract must be generated to create a contract between seller and buyer. Some of the high-level languages used to create smart contracts are as follows; Solidity, Vyper, and Bamboo. However, smart contracts created with these languages cannot be run directly on Ethereum nodes. First, to enable communication between the contract and the node, which is an abstraction level, the contract must be understandable to the Ethereum Virtual Machine (EVM), which is written at a low level. High-level languages like Solidity and compiled to low-level machine instructions which are opcodes. Running the created opcode, EVM allows the smart contract to access the Ethereum nodes.

In addition, this opcode is used not only for EVM but also for creating an Application Binary Interface and gives a more readable json output with all inputs and outputs. The owner must pay gas for each byte EVM reads. Therefore, smart contracts should be written in the most optimized way and loops should be avoided [16]. In the designed loyalty system, the main smart contract is optimized by providing gas controls of the functions with many smart contracts and the cost of generating tokens is equal to 0.000037 Ether.

Using real ether while developing is very risky because an incorrect block of code can cause large amounts of financial losses. For this reason, there are test Blockchains that work with different consensus algorithms. There are 3 different Ethereum network types which are Rinkeby, Kovan, and Ropsten. The Ropsten network works with the proof of work consensus algorithm and is the network closest to the main network and the ether can be mined easily in this network. Kovan and Rinkeby networks work with proof of authority consensus algorithm and ether is not mined, only requesting is enough. Smart contracts working in such networks can be easily followed and are visible to everyone. Even detailed information such as tokens in total and which addresses they have been sent to can be accessed [24].

In addition to the online test networks, it is also possible to create a local chain. In such networks, there are 10 ether addresses by default and 100 ether are created in each address. Development can be done easily on these and there is no need to make any requests or mining. In addition, with Remix, the Solidity code can be written in the desired version via the web browser. Remix is a website that has a connection with a local chain with web3.js infrastructure. The smart contract can be programmed on the site without any need to install any software and even suggestions for optimization can be given. Due to the debug feature provided by the remix, it is the most preferred online Solidity compiler during the development of the smart contract [25].

2.3. Loyalty Programs

Manufacturers and markets have been developing and implementing the loyalty system since the late 19th century to encourage customers to shop. S&H Green Stamps

program launched the first modern loyalty system and started to distribute green cards to other businesses. This loyalty system gives customers a green card as a reward for shopping and thus, different products can be bought by using these cards. In the early 1980s, loyalty systems began to be used by airline companies and became quickly popular [26]. In 2014, loyalty programs reached a total of 2.4 billion members. In 2016, the number of members using the loyalty program reached 3.8 billion. The distribution of loyalty programs for 2017 by sectors is shown in the pie chart [4].

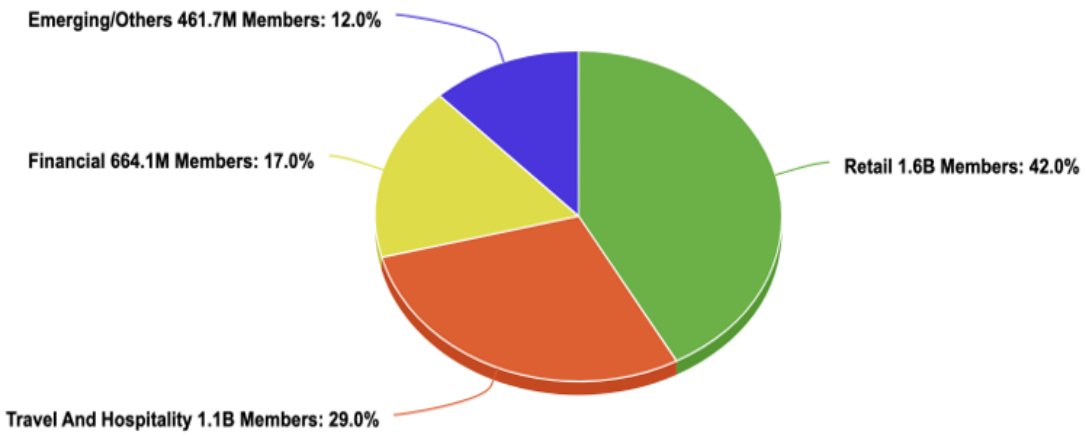

Figure 2.4. Loyalty Programs According to Different Sectors [4]

There are 6 traditional types of loyalty programs which are basically as follows; Punch Cards, Points, Tiered, Fee-Based, Cash Back, and Coalition loyalty programs. In the Punch Cards loyalty program, there is a specific card belonging to the customer and it is punched every time when goods or service is purchased. When the number of holes in the card reaches a desired number, the customer can use the card to get a free product or service. This type of program is the simplest loyalty program and is implemented by most cafes and restaurants nowadays.

In the Points loyalty program, each shopping activity of customers earns them points and these points can be spent to get a service or product afterwards. Nowadays, this loyalty chain is tried to be created by giving points to customers entering a system, or in exchange to following their social media accounts. Virtual cards of supermarkets and gas stations have the Point Card loyalty program. For instance, when 20 liters of

gasoline is purchased, 200 points is loaded on the card to be used for the upcoming purchase.

The Tiered loyalty program is very similar to the point system as a reward system, but here points are earned according to the seniority of the customers. Green and gold members of various coffee chains can be given as an example of this customer seniority. It is ensured that the customer constantly spends a large amount of money in the said system, thus making it difficult for the customer to leave the loyalty system.

In Fee-Based loyalty programs, the customer pays a specific amount of money to be included in the system. The customer within the system can benefit from the service that he or she has paid unlimitedly. For example, when customers pay a monthly membership to an online movie platform, they can watch as many movies as they want in the system.

Cash Back loyalty programs make product-based or customer profile-based cash back so that the number of customers of competitors can be reduced or the sales of products with fewer sales can be increased.

Coalition loyalty programs are the more attractive type of loyalty where customers can collect their rewards. It is a system preferred by most big companies. Flight miles cards of airline companies can be given as an example [27].

Today, with the increasing importance of the loyalty systems, many studies are carried out on the feasibility and transfer of these systems to the Blockchain. These studies are examined in detail in Chapter 3.

CHAPTER 3

LITERATURE REVIEW

There have been many studies carried out to apply Blockchain technology to the loyalty systems. Some of the more relevant ones are summarized in the proceeding paragraphs.

The article in reference [28] investigates the effect of Blockchain technology on loyalty systems. With the help of self-determination theory, the needs in the fields of economy, autonomy, competence, and relatedness are defined. The results are categorized into three dimensions: economic utility, psychological self-fulfillment, and social interaction. The name of the tokens used in the established theoretical loyalty system is 'Yun point' and these tokens are purchased from a platform that holds the mint authority called 'gege point'. Companies and customers can transact with the generated token and transfer among themselves but must pay the minter to use the tokens. When the customer buys a service or product, the market gives some 'Yun tokens' and the customer can purchase products from another market using these tokens. For self-determination theory, it has been determined that it meets the requirements in the specified areas. The economic requirements it meets are as follows: customers were able to buy the product they wanted to buy thanks to the tokens they accumulated and did not pay extra money for this, so the economic benefit was observed. If autonomy requirements are mentioned; as customers can easily exchange the tokens they earn digitally with desired services, they perceive value of self-fulfillment. If competence is mentioned, the needs of customers are met in this field since they can use the tokens they earn through a single interface. Relatedness requirements were met as the customers could share the tokens in their wallets with other customers. It has been suggested that the system designed as a case study outcome can increase the customers of the companies. Since the system set up in the article is theoretical, there is no Blockchain network, standard or smart contract information [28].

In [29] publication review processes are said to be quite long and progress with low efficiency. Moreover, it is claimed that a few have been done to eliminate this chronic

inefficiency. Today with the advent of Blockchain technologies it is mentioned that the peer review payment system, which is a token-based system, can be a solution for the mentioned problem. In accordance with the smart contract between the author and reviewer with a token-based payment system, the author gives 100 peer-review coins (prc) to the system and the system receives 1 prc for this continuity and distributes 33 prc to 3 reviewer teams. The good evaluation of the reviewed article and its completion in a short time in terms of publication review processes can increase the earned token. The earned prc tokens can be converted into tradable money on the stock market, but also the increase in the amount of the token earned by the reviewer corresponds that it will be preferable for reviews of articles. Thus, incentives are created for reviews and editors. While a system that works this way can facilitate market regulation of peer-review coin, it can also be designed to provide some new features that existing publishing ecosystems cannot provide [29].

In the mentioned system in [5], a loyalty strategy was defined for students and staff to operate with the selection of 12 retailers. While the user is operating in the system, the user reads the barcode defined from his phone and then sends the pico user ID and timestamp information to the cloud. The information in the cloud reaches the Ethereum node by using the platform developed by a predetermined company. The loyalty system tested at the University of New South Wales (UNSW) works by converting the collected stamps into the ether. If the customer has accumulated 10 stamps, the loyalty platform requests ether from the Ethereum chain and the taken ether is loaded into the customer's account. Users in the system gain digital stamps as they read the barcode and can up to gain 5 stamps per day. In order to keep the participants longer in the loyalty system, marketing strategies have been determined. There are promotions such as double win campaign in the second week and a $ 5 ether gift for the first registration in the system. The customers can easily earn cryptocurrency by conducting a business transaction or answering marketing questions and can easily use the amount they earn whenever they want. Prior loyalty systems such as points/miles have reinterpreted with a new perspective and results are obtained by a survey. The questionnaires and satisfaction levels of merchants who are included in the loyalty system using the Ethereum chain have mentioned at UNSW Sydney campus. Although the system differs in technology and operation, it was shown that the loyalty chain used in conjunction with Blockchain technology has a win-win relationship and can quickly become popular. In addition, 21%

of users are logged into the system but have never made any purchase, 18% of them earned 1 to 9 stamps, and 61% earned 10 stamps or more. Moreover, according to the survey results, 86% of people have found the idea of earning ether from scales appealing and 67% indicated that they prefer to keep ether instead of cash. Although it is not exactly similar to the created system in this thesis, newly created features resemble. The new generation loyalty system, which is formed by the combination of Blockchain and loyalty token, has been attended by 177 people and the survey results show that people tend to choose the next generation loyalty system [5].

National Taiwan University of Science and Technology produced NTUSTokens on a private Ethereum chain in order to increase students' technology awareness and on-campus activities [30]. A private chain is preferred because the users log into the system with their student IDs and the chain only works inside the campus. Users can earn tokens by going to the library or doing sports, and even send these tokens to their friends and as a result, they can quickly adapt to campus life. Moreover, paying attention to the use of electricity or consuming water resources consciously enables them to earn tokens. On the other hand, the negative behaviors of the users are processed into minus tokens and are punished in a way. Since it is a system established in 2017, the Solidity version used is outdated and no existing standard was followed while the token was being produced [30].

Another loyalty system shows resemblance to the created system in this thesis, but the Blockchain-based cross-organizational integrated platform has not inherited any standard, and the Solidity version is also outdated [31]. According to the article, goods providers and token providers can register to the system and put the desired product with the desired amount of token and users can shop with the tokens as much as they can. The good providers within the established system have a private chain among themselves. They also can access the database and users with the webserver called BCOIP. The installed system functions like a Blockchain-based shopping site. It was also mentioned that the established system is suitable for the operation of shopping malls and airline companies [31].

The loyalty system desired to be established in [32] was established on the neo chain and tokens called Promotion Asset Exchange were produced. In the installed loyalty system, customers can earn tokens by reading the QR codes of the products they buy with their mobile phones. The mobile phone connects to the neo Blockchain with neon.js and requests a token. Customers can use the collected tokens to buy products from merchant

and the merchant accumulates the tokens they receive in return for the product they give to the customers and gets money from the contracted manufacturer in exchange for the token. The system objectives mentioned are similar to those of the system we have proposed, but the used Blockchain technology and the operation of the system are completely different [32].

Table 3.1 gives a summary of the studies on loyalty systems and Blockchain. Each study was compared according to the token type, blockchain technology, token flow, platform, classification and the target of the research articles.

Table 3.1. Comparison of the Proposed Loyalty System with the Literature Studies

Author Name	**Token Type**	**Blockchain Technology**	**Token Flow**	**Platform**	**Classification**	**Target**
Wang et al. (2018)	Yun Point	Not mentioned	Different	Website	Theoretical	Customer Loyalty
Avital, M. (2018)	Peer Review Coins	Not mentioned	Different	Website	Theoretical	Publication review processes
Shelper et al. (2018)	Stamp usage (non-token)	Ethereum	Not similar	Mobile phone	Practical	Customer Loyalty
Díaz et al. (2017)	NTSUSToken	Ethereum	Different	Website	Practical	Increasing student awareness
Liao et al. (2019)	Reward point	Ethereum	Not similar	Website	Practical	Cross or-ganizational integrated platform
Bülbül et al. (2018)	PAX Token	Neo Chain	Similar	Mobile phone	Practical	Customer Loyalty

CHAPTER 4

PROPOSED LOYALTY SYSTEM

4.1. The New Generation Loyalty System

Loyalty systems have tried to include customers in many ways until today. But customers no longer prefer to be in classic loyalty systems. In traditional loyalty systems, a card needs to be carried physically and one type of service can be obtained for the earned payments. For instance, when customers use air transportation, a certain percentage of the distance is given back to the loyalty card as a reward and the customer can spend the earned miles before the expiration date of the next flight. However, customers cannot convert the earned rewards to currency in such systems, only one type of product or service can be purchased as a reward. Also, customers in traditional loyalty programs forget their eared tokens, reward or points because of difficulty to follow them.

Our contributions to the traditional system are as follows. The manufacturer can create tokens in the desired amount, can make the created tokens available to the stock market, or just keep them in the loyalty system. Just like cryptocurrencies, the produced tokens can be transferred between wallets. Customers in the system do not have to carry different cards for different loyalty systems of each company, an Ether address will be enough. The customer can earn the IZTECH Tokens produced by the manufacturer, as they purchase products from the contracted markets. Moreover, depending on the strategy of the manufacturer, customers can buy the token from the stock market or exchange it with cryptocurrencies. Users can purchase services or products by giving IZTECH Token to any market included within the system instead of cryptocurrency. Unlike the above given flight example, users can benefit from their rewards not only in their next flight, but also when they buy coffee or sandwich. In the loyalty program designed as explained in the example, the loyalty of the users within the system is increased as the users can spend the tokens and rewards, they earn according to their own

wish. One of the important advantages is that users can convert these Ethers in the stock market. Even the gained tokens may increase in the value of the stock market as the supply-demand increases, and the tokens that stay in the customers' accounts can gain value. This gives customers the opportunity to earn much more than they spend, not just to recover some of what they have spent.

The loyalty system built on Blockchain is accessible and transparent to everyone, and thus the customers can follow their earned tokens. In addition, it is useful to use Blockchain not only for the customer but also for the manufacturer. Because, according to the manufacturer's market strategy, token production and earnings balance can be determined easily. Ethereum chain is considered to be the most suitable and usable Blockchain network for the system and improvements have been made on this chain.

The tokens used in our loyalty system are named as IZTECH Token. These tokens were produced on the Ethereum chain by following the standards which are derived from the ERC20 [33]. The Solidity programming language was used to write smart contracts. This language, which is formed by the blend of object-oriented, C++ and JavaScript, is compiled in Ethereum virtual machine and run on the Ethereum chain. Each transaction written on Solidity has an operating fee on Ethereum. Therefore, smart contracts were created taking the amount of gas consumption into account and avoiding loops.

The main contract responsible for the production and control of IZTECH Tokens is the combination of many libraries and contracts. Smart contracts can be considered as classes, as in object-oriented languages, and these inherited functions can be used within the parent contract.

The ERC20 standards in the library have been followed and new features have been added on them and the created tokens are named as IZTECH Token. So why the token was produced, could ether or any cryptocurrency be used instead of tokens? The loyalty system uses tokens because fewer resources such as time, currency and energy are consumed compared to the generation of all cryptocurrencies. In addition, these difficult-to-obtain cryptocurrencies cannot be canceled or undone after they are issued to the user. They cannot be produced and delivered to customers quickly according to the supply quantity because it takes a long time to obtain.

Another important point is that an error in smart contracts can cause high-cost loss in transactions with cryptocurrencies. Because of these disadvantages, tokens were preferred to be used. It is almost costless to produce tokens because they act as an object

in the network and numerical operations can be done easily on them. Another advantage is that the given tokens can be destroyed if the customer owns an excessive amount of token in a short time period or gain tokens through illegal ways. If the supply and demand increase sharply and it becomes necessary to generate tokens, the owner of the smart contract only needs to run one function for mint.

Figure 4.1 illustrates IZTECH Token Flow between the manufacturer, market, and the customer.

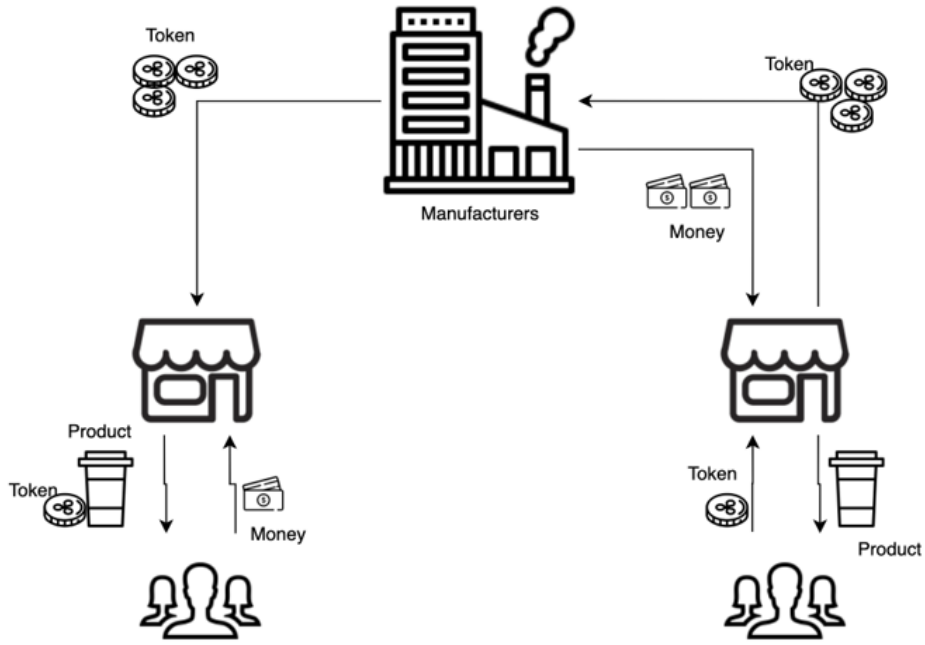

Figure 4.1. IZTECH Token Flow in the Designed Loyalty System

The sequential token flow in the proposed loyalty system can be seen in Figure 4.2 on the next page.

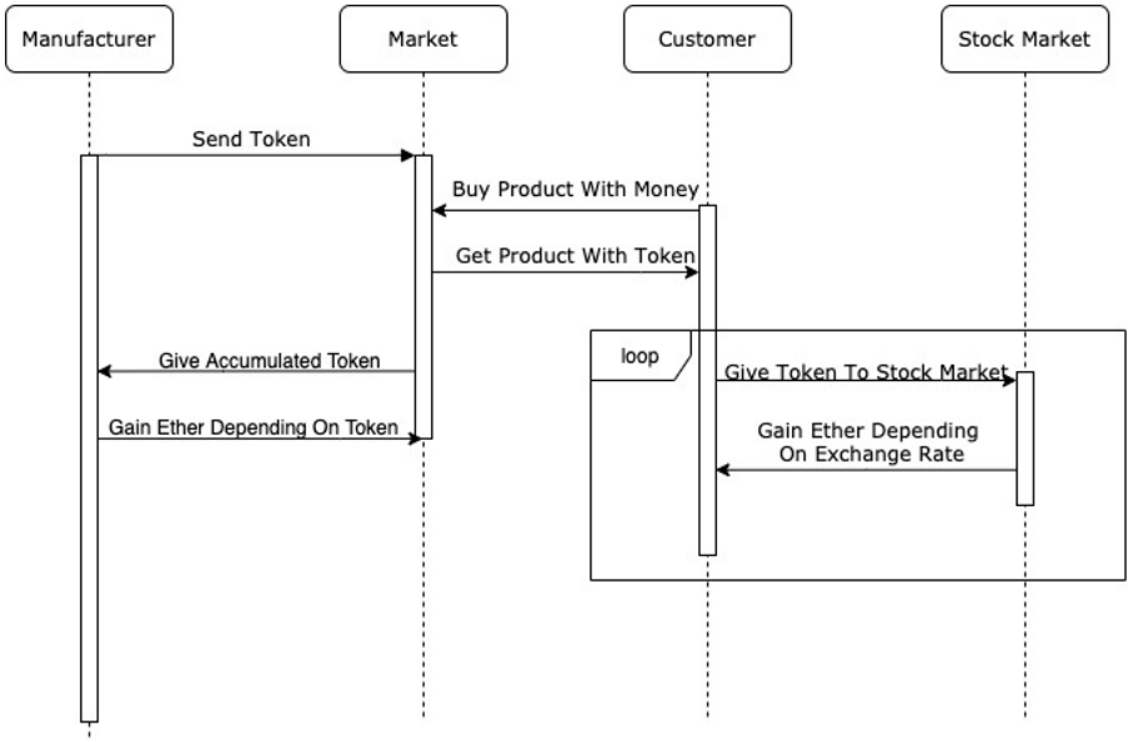

Figure 4.2. Sequential Diagram of the Token Flow

At the first stage, the manufacturer, who has a contract in the designed system, starts sending the tokens to the markets in the loyalty system. Markets receive tokens instead of money from customers for their products and services, as customers can get products and services from the markets with the tokens they have accumulated. In addition, customers can exchange token to Ether from the stock market. Moreover, the accumulated tokens are given to the manufacturer and the manufacturer gives Ether to markets according to the accumulated token amount in the markets to a determined rate. Thus, the contracted markets make profit in the system.

The more the customer spends the more the tokens gain value and as tokens gain value, the customer tends to spend more in the designed system. As a result, the customer continues to remain inside the loyalty system as a user. Through this, the manufacturer can produce more services and products and make agreements with more markets. The markets within the system can sell more products and services as users shop and earn ether back. Since customers earn as much as they spend, they are included in the system and are encouraged to spend more as they stay in the system.

All token transactions and transfers can be monitored via the Rinkeby network [34]. A web interface was necessary for the manufacturer to perform arithmetic operations. The general structure of the loyalty system can be seen in Figure 4.3.

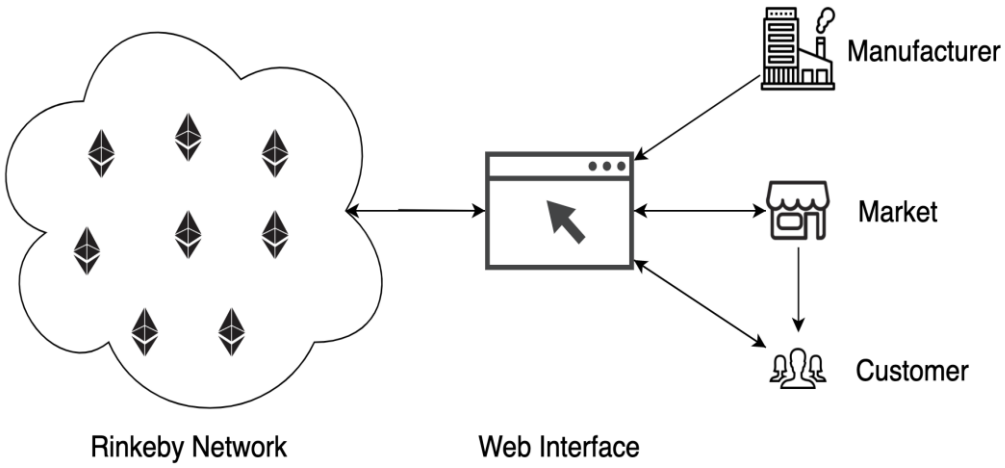

Figure 4.3. General Structure of the Loyalty System

Rinkeby network is connected to web interface with Infura. Web interface consist of the combination of react.js, web3.js and lots of npm packages. and web interface is used by manufacturer, market and customer. The react.js library was used for the web interface. Because react.js could not directly access the Ethereum node, web3.js was integrated into react.js.

It was not preferred to run the new generation loyalty system on the Ethereum main net, because the ethers traded on the main net are not as easy to produce as on the Rinkeby network and are produced with consensus algorithms by the miners. However, in order to produce ethers in the Rinkeby network, ethers are requested only with Faucet [35].

Because the amount of expense of customers is directly proportional to the amount of gain, not only the continuity of the customers within the system is provided, but also the manufacturer and the market make profit. As a result, the targeted win-win-win relationship in the new generation loyalty system is established.

4.2. Technology

As mentioned before, in order to deploy the smart contract on the Ethereum chain, it must be written to the EVM in a runnable language. For now, Solidity version 0.5.0 is the most reliable and stable version of the smart contract development tool. Tokens can be produced in desired ways and make it more acceptable to users as it means that it is safer to adhere to a standard. The Openzeppelin library, which has been tested by many users and written with the highest level of code quality was used for this purpose [6].

Truffle was used to compile the smart contract and deploy it to the Ethereum chain Truffle is also convenient for changing the Solidity version of the written smart contract. In addition, local Blockchain was created to deploy smart contracts in the development phase by using Ganache and transaction tracking was performed from the Ganache application interface [36, 37].

To communicate with the web interface of the established network, the beta version of the web3.js library was used and implemented on the React.js library Meta Mask, which is a Google Chrome extension, was used for tracking the generated tokens and for permissions during spending [38,39].

After local development was completed, a real Ethereum network called Rinkeby network was used. The new deployed smart contract on Rinkeby was used for transactions and the feasibility and maintenance of the tokens were checked on the network.

Infura and hdwallet provider was used for deployment. Infura is the gateway to the Ethereum node in a way that allows the smart contract to access the desired the Ethereum network [40]. In order to access Ethereum account, the hdwallet provider which is npm package was used and mnemonic information that is uniquely defined in each Ethereum account should be given as an input.

4.3. Development

Before starting to set up the system, it was searched which chain would be more suitable for the desired system. Research has shown that the development of the Ethereum

chain is done more quickly since there are already many tools developed. In addition, the high number of transactions per second is an important factor in the selection of the Ethereum [41]. In order to make transactions on the Ethereum chain, a smart contract was written to meet the requirements of the loyalty system.

The token types such as ERC20, ERC721, and ERC77 on the Ethereum chain are derived from the standard Open Zeppelin library. ERC20 token standard is used in the established system. The functions that have been added to the smart contract have determined the working methods and rules of the award system. In addition, since it requires gas to be used for each process, the loop functions were not used in the project and the problem was solved by mapping where the loop operation was required.

IZTECH Token consists of a combination of smart contracts, two libraries, five smart contracts and an interface that enables the production of tokens as can be seen in Figure 4.4.

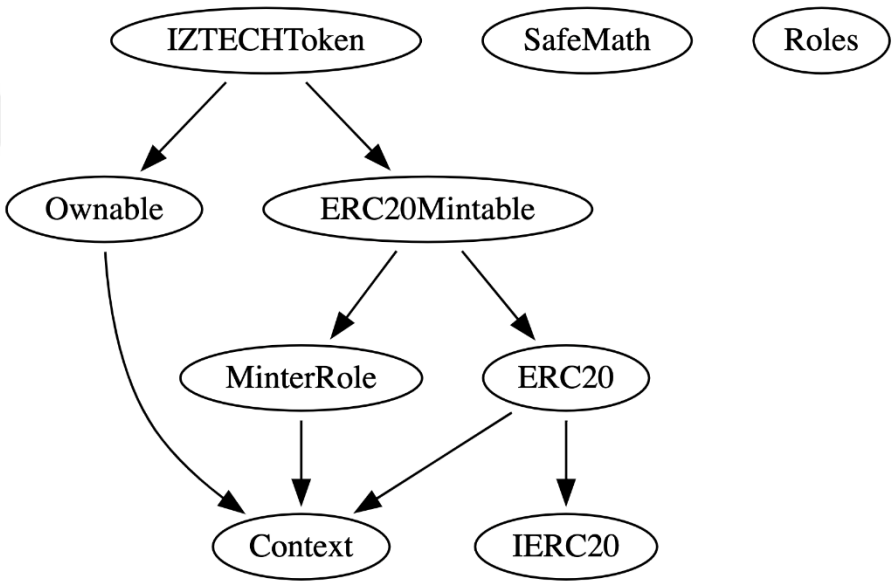

Figure 4.4. Inheritance Relationships of Smart Contracts

Safe Math library prevents overflow errors during arithmetic operations and limits the use of gas. The Roles library assigns the role to the given addresses and has 3 basic mapping functions. It can only use the functions of the desired contract with ‘using’ keyword. The Context contract, which is the most commonly used smart contract, was used to prevent direct access to owner information and data. The MintRole smart contract

was produced by inheriting the context contract and the Role library was used. The MintRole contract is the contract where transaction authorizations are created on tokens, and IERC20 is the contract where the standards of tokens are determined and were used as an interface. In IERC20, there are functions that need to be overloaded in the parent contract. ERC20 contract adds functionality by overloading the functions specified in the interface. Also, token arithmetic operations were performed using the Safe Math library in the ERC20 contract. The ERC20Mintable contract is derived from the MinterRole and ERC20 contracts, and it is the contract in which the mint function is defined. The ownable contract contains functions that the owner of the contract can use. Thanks to this, the owner of the contract can delete his or her own role or can transfer to other addresses. IZTECH Token is made of Ownable and ERC20Mintable contracts and Safe Math library is used for the arithmetic operations. Finally, with the combination of smart contracts and libraries created in a modular structure, usable IZTECH Tokens are produced within the system. Thanks to this modular structure, not only the code quality and security of the smart contract spent have been increased but also the gas consumption has been reduced additionally.

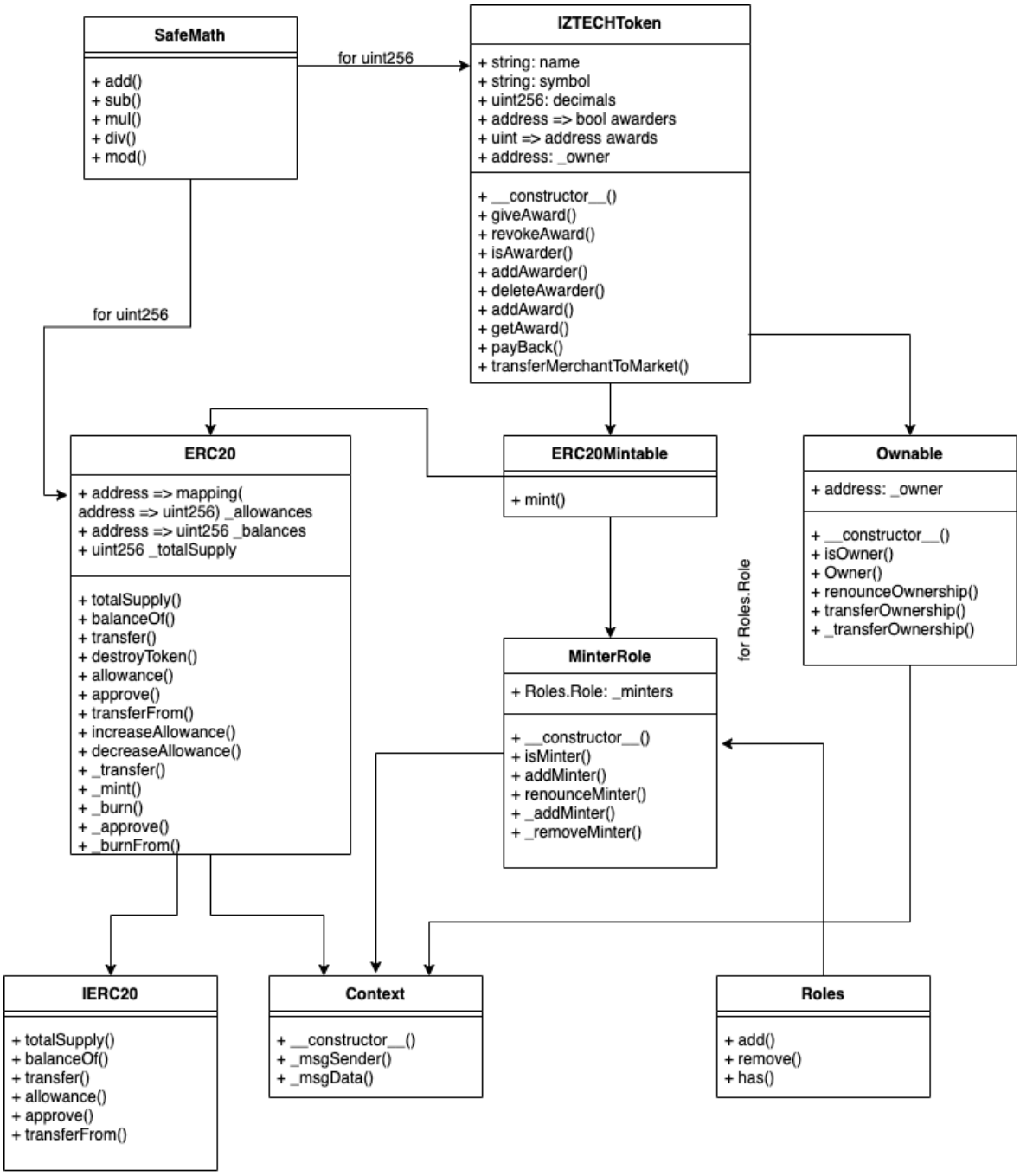

Figure 4.5. The Architecture of IZTECH Token System

The dependency of the contracts created in the study can be seen above. The "context" contract is required for Ownable, MinterRole, and ERC20 and is transferred inherently to these three contracts. Context contracts enable data such as msg.sender and msg.data, which can be accessed directly in Solidity, to be accessed by internally generated functions in context. Msg.sender and msg.data should not be used in a smart

contract derived from this contract. Instead, the assignment should be made with the functions in context.

The ownable contract is a contract produced using context and its main task is to create a basic access control mechanism. This contract creates a modifier called OnlyOwner, and nobody, except the contrast owner, can run the functions that use this modifier. In Solidity language, modifiers change the body of the function and check if it meets the desired rules, that is, it works before the function it is used and provides the opportunity to make the desired changes. When the ownable contract is first run, the msg.sender function defined in the context is executed and the result is assigned to the owner variable. The owner of the contract is now initialized. In addition, the owner function returns the owner's information. The IsOwner function checks if the address that calls the function is the owner and returns a boolean value. The RenounceOwnership function is a one-way function and is run when the owner wants to leave the title of the contrast owner. After this function is executed, no method with the onlyOwner modifier can be run. Finally, the transferOwnership function is called when the owner wants to move his status to another address.

It is the library used to assign the role contract to the addresses. The role library does not need to be inherited with "is" when creating any contract. It is sufficient to state that the variable is created, and the role library is used. This library basically performs mapping and has 3 functions that are "add", "remove", and "has" for role control.

The MinterRole contract is inherited from the context contract and uses the role library. This contract is the contract in which the mint authority of tokens is checked and when it is executed, the addMinter function in the constructor runs and sends the msg.sender address from the context contract to the role library and authorizes it. The onlyMinter modifier exists to be used later and to provide mint authority controls. As the public function, the isMinter function checks whether the given address is minter and addMinter gives the authority to mint the given address. Thanks to the renounceMinter function, the minter can give up its minter role authority. If enough tokens have been produced on the network and there will be no changes in the market strategy, this function can be activated, and the mint authority can be disabled.

The IERC20 interface is the place where all functionality and the main structure of each token is determined. In this contract, the functions to be used in ERC20 tokens and their inputs and outputs are specified. These functions include as follows; totalSupply

returning the entire token amount, balanceOf returning the token amount of the given address, and transfer function, that takes the address of the recipient and the token amount to transfer tokens between accounts.

SafeMath library is a library that increases security by protecting the operations we use frequently while working on tokens against overflow. While performing these operations in Solidity can consume all the remaining gas, through the SafeMath library, an error occurring in opcode does not consume the remaining gas amount. The arithmetic operations that can be done using this library are added, subtraction, divide, multiplication and mod.

The ERC20 contract, which is the contract established on libraries and interfaces and token transactions were created by using context, IERC20 and SafeMath contracts. Functions that were inherited from the IERC20 interface were initialized in the mentioned contract. Additionally, internal mint and burn functions were added and through these functions, tokens can be generated, and the produced tokens can be destroyed.

The ERC20Mintable contract, which combines two separate contracts where tokens can be produced and the authority to produce is controlled, includes the mint function and thanks to this function, tokens can be produced. The onlyMinter filter from the MinterRole contract must be passed to run this contract, which indicates that not every user can mint the token. This function takes both the address and the token amount and returns a boolean value.

As a final step, the IZTECH Token contract combines the contracts created and includes modifications according to the desired market strategy. The name, symbol and decimal value of the token which is used to compile the smart contract for the first time was created. Then the address was mapped to the boolean and its value was assigned to the awarder's variable, while unsigned integer was mapped to the address and its value was assigned to the awards variable. At the end of these operations, there is the constructor method that mints the token for the owner of the smart contract. The giveAward function first checks whether the token to be given belongs to the owner of the contract and then checks if there is sufficient balance for the transaction. If it fulfills all the mentioned requirements, it performs transfer with superclass transfer function. The revokeAward function was used to disable previously given tokens. In order to run this function, the contract must be owned. The isAwarder function checks whether the given address is authorized to give tokens. The onlyAwarder function is run as a modifier and

checks the role of the given address. The addAwarder function allows dealers in the system to assign tokens. The deleteAwarder function disables the dealers' ability to issue tokens. The addAward function separates the token types by enumerating the generated tokens. PayBack function is the function that gives dealers ether based on the number of tokens collected by the dealers. This function takes the amount of token address of retail and exchange rate and sends the ether to the retail address. Since currency transfers on the Ethereum chain are carried out over the Wei coefficient, the ether sent during the calculation is multiplied by the Wei coefficient and divided by the exchange rate. The TransferMerchantToMarket function is used to send a large number of tokens to the markets, and the tokens produced by the merchant cannot be given to markets as a reward, so they are carried out by another transfer outside the award system.

The installation of the system can be divided into 3 main phases. The first stage is to create and test the smart contract online compiler, then to migrate the written contract and run the web interface on the local chain, and finally run the written contract on the Ethereum Rinkeby chain.

In the first phase, Solidity was learned to write the smart contract. As Solidity is a language that changes as a new vulnerability is available, older sources cannot completely help. Therefore, the documentation of Solidity has been analyzed [42]. Afterward, researches were conducted on how to transfer the information about the loyalty chain to the smart contract with Solidity. Finally, after determining the market strategy, smart contracts were written. At this stage, the documentation of the Openzeppelin library is read and the necessary contracts are integrated into the system. Since the contracts were written, from the smallest contract to the larger contract in terms of functionality, the solution of the errors encountered has become much easier. Contracts do not need to be rewritten to add new features to the created loyalty system, as the new feature can be transferred to contracts created as an inheritance. The online compiler, which is remix editor, was used to test the contract written for the first phase [25]. Developments were made in 10 ether accounts prepared for the test. Using the online compiler, the smart contract was converted to bytecode with the latest version of Solidity and optimized to minimize gas consumption.

After the first stage of completion, the smart contract was deployed to the local chain. In order to create the local chain, Ganache was used which generates fake blocks and puts the transaction information in the blocks, thus creates a chain in the desired port.

A smart contract was deployed to the created fake blocks using truffle, that compiles the smart contract by looking at the config file and sends the generated bytecode to the Ethereum virtual machine. The frontend written using React.js was connected to the Blockchain running locally and therefore, the loyalty system has started working on the local chain. Web3.js has been integrated into the system in order to enable the frontend to communicate with the Blockchain. At this stage, the development of the frontend was emphasized. The frontend was developed with the structure of the components. The Application Binary Interface, which is created after the smart contract is compiled, is integrated into the React.js and called with function names and inputs from web3 are given. The currentProvider function in web3.js was used to allow the inputs of the functions to access Blockchain nodes. However, in the beta version of the web3.js, installing and using web3.js with the node package manager can be problematic. When the wallet is changed in MetaMask, it automatically takes the information of the wallet used in the reactive application. This allows the smart contract owner to mint the tokens and sends them to the dealers. At the same time, users can transact with these tokens.

After the second stage, the system was deployed to the online Rinkeby chain, in which anonymous people from all over the world can control the flow of tokens across the network. An application programming interface (API) number has been created for deployment that can be connected to the Rinkeby network node by getting a free membership via Infura. The config file of the Truffle has been changed so that the contract can access the Rinkeby network. Also, the hdwallet provider npm package was installed and this package was used to access the user's ether account. Thanks to the config file, Truffle first accessed the ether account and then deployed the contract to the Rinkeby network through the accessed account. In addition, Infura can be used not only for the Rinkeby network but also for Ethereum networks, that is, main net can be reached with the same config.

Table 4.1 shows the ERC20 token flow on the smart contract. It is taken from Rinkeby network and can be seen by anyone. In this table, there is information about the transaction hash block number, Unix timestamp, DateTime, sender address, receiver address and quantity of token. All transactions on the main contract, which is (0x2a971930703e427cf64f4850b2f656f2e67fd014), can be seen on the next page.

Table 4.1. ERC20 Token Flow on the Smart Contract

Txhash	Blockno	Unix Timestamp	Date Time	From	To	Quantity
0x3a47b5a6be2200e1c3acb2841f46f45b29451fd722eee655b5576bbdec468b2e	6246370	1585849793	4/2/2020 5:49:53 PM	0x0000000000000000000000000000000000000000	0xaaba235fb01f054f596f62798efb762b899f93c3	10^5
0x762a0639b3a87a50ad952a8c3ba00486880add76e7e803864a98f49fbb09dd51	6246377	1585849898	4/2/2020 5:51:38 PM	0xaaba235fb01f054f596f62798efb762b899f93c3	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	222
0x1683cd485cfa13c25b9e74ea3c5d71607480d08b6add7487cb296cbdd72c15a6	6246381	1585849958	4/2/2020 5:52:38 PM	0xaaba235fb01f054f596f62798efb762b899f93c3	0x3cb2fac5351fe7dd5bb4b4e4ad8775cc2f312e2c	222
0xb5f083eea7e9ef81a71f866b6e88b0c4561bdc643b65954b09478b7be3326c95	6246384	1585850003	4/2/2020 5:53:23 PM	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	0x0000000000000000000000000000000000000000	222
0x6f6873623a5fcb08d01a2e2e16d808db913b6c1d0f6d1daa68852c4fb26c2c28	6246389	1585850078	4/2/2020 5:54:38 PM	0xaaba235fb01f054f596f62798efb762b899f93c3	0x6c4f653f8454d856f737c2cb38cef7e1ad1c8e00	1,234
0xa1ee9439ed1070a9cd04e8f04ae0775debe083a98bb19f3c0c127eba68e2fffb	6247087	1585860553	4/2/2020 8:49:13 PM	0xaaba235fb01f054f596f62798efb762b899f93c3	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	5,321
0x95263a04eb2569d4df1cbef31ad87103da4bf39015a287323e30d210db8ec542	6247088	1585860568	4/2/2020 8:49:28 PM	0xaaba235fb01f054f596f62798efb762b899f93c3	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	5,321

All transactions of the selected addresses can be seen in Table 4.2. The table gives information on when the transactions were made, on which smart contracts it was made, the sender and receiver addresses, and the amount of tokens traded. For this reason, users in the new generation loyalty system can easily follow the awards won. Last transactions belong to our last version of main contract and information of other transactions can be seen below. As can be seen, many different contract addresses exist in table because different contracts in development phase were compiled.

Table 4.2. All Token Transactions on a Wallet

Txhash	UnixTimestamp	DateTime	From	To	Value	ContractAddress	TokenName	TokenSymbol
0xbf3ca41291fc232257c818ed7376a5b7be7bee0293b43957a6d5d632e50f0514	1585841826	4/2/2020 3:37:06 PM	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	0xaaba235fb01f054f596f62798efb762b899f93c3	1,111	0x37ff2c80b15211424fde8848518f3e1d91b20ccd	IZTECH Token	IZTECH Token
0x563709784517bc1667ae7d04352b941221a416e1f5f15bdea3f8f9825fca8d5c	1585842066	4/2/2020 3:41:06 PM	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	0xaaba235fb01f054f596f62798efb762b899f93c3	2,222	0x37ff2c80b15211424fde8848518f3e1d91b20ccd	IZTECH Token	IZTECH Token
0x75f9cf8d1e4b80a35965fabac0aa430f5b62feb26c892356fee5007967cf2d78	1585842966	4/2/2020 3:56:06 PM	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	0xaaba235fb01f054f596f62798efb762b899f93c3	333	0x37ff2c80b15211424fde8848518f3e1d91b20ccd	IZTECH Token	IZTECH Token
0x6ff3fab49a71f04bc3d94947be8f36d7c191704d8f00fe79c6c841db66cd5c40	1585844768	4/2/2020 4:26:08 PM	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	0xaaba235fb01f054f596f62798efb762b899f93c3	1,111	0xd984588f02354a0c1b61ca3c4bb2646de42acbcb	IZTECH Token	IZTECH Token
0xa87c0e65128d8a2a2404570d2db96734fdb6a394d7215119c067309c4370d472	1585844828	4/2/2020 4:27:08 PM	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	0xaaba235fb01f054f596f62798efb762b899f93c3	1,111	0xd984588f02354a0c1b61ca3c4bb2646de42acbcb	IZTECH Token	IZTECH Token
0x65be72eb54fd039fa404a65fe6932238e6d86ac1458e8b258d76895c28888ff2	1585845413	4/2/2020 4:36:53 PM	0x0000000000000000000000000000000000000000	0xaaba235fb01f054f596f62798efb762b899f93c3	100,000	0x91161de18c8fbedcddc3635cd412f0bfea34d1b6	IZTECH Token	IZTECH Token
0x4699d76b28e8cf693ca0dbe3ff69f0b3ee60abe0e79a88e08b0a9c2e02ce22a5	1585845458	4/2/2020 4:37:38 PM	0x0000000000000000000000000000000000000000	0xaaba235fb01f054f596f62798efb762b899f93c3	1,111	0x91161de18c8fbedcddc3635cd412f0bfea34d1b6	IZTECH Token	IZTECH Token
0x3264bcb316fc22fdd35462a1f09b5d8a8cb3ad033a5c9f532b59e1d20c089efa	1585845518	4/2/2020 4:38:38 PM	0xaaba235fb01f054f596f62798efb762b899f93c3	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	1,111	0x91161de18c8fbedcddc3635cd412f0bfea34d1b6	IZTECH Token	IZTECH Token
0xb6fbab49244baa67d9ef2d9562ac932b521e3a5ee2b1d1a2e67b15b20458f154	1585845593	4/2/2020 4:39:53 PM	0xaaba235fb01f054f596f62798efb762b899f93c3	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	3,333	0x91161de18c8fbedcddc3635cd412f0bfea34d1b6	IZTECH Token	IZTECH Token
0xfc5d9708c620c741f985ee69d99d16cc5c8c7ebccdb8935a846ec6a4e051079e	1585845683	4/2/2020 4:41:23 PM	0x0000000000000000000000000000000000000000	0xaaba235fb01f054f596f62798efb762b899f93c3	100,000	0x70f3a572df64e3655eda06051d96151ead4f7db9	IZTECH Token	IZTECH Token
0xfb15f776b3ce3d9d2e23ea4e4f25a12dcf644220952a1cc85bd288464b0dfb7c	1585845728	4/2/2020 4:42:08 PM	0x0000000000000000000000000000000000000000	0xaaba235fb01f054f596f62798efb762b899f93c3	22,222	0x70f3a572df64e3655eda06051d96151ead4f7db9	IZTECH Token	IZTECH Token
0x8a0d303945e8d5869ecdd49f057d2c161c2bd60dc2f2d95a97d48de893577b69	1585845833	4/2/2020 4:43:53 PM	0xaaba235fb01f054f596f62798efb762b899f93c3	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	4,444	0x70f3a572df64e3655eda06051d96151ead4f7db9	IZTECH Token	IZTECH Token
0x3129406a21fbd7c05a9b9f92d709a6c47b980e7a87b97a924ff96462da551a98	1585846748	4/2/2020 4:59:08 PM	0x0000000000000000000000000000000000000000	0xaaba235fb01f054f596f62798efb762b899f93c3	100,000	0x30b216dfa8548dbb76d6abac892d8aec35fbf550	IZTECH Token	IZTECH Token
0xe6b04ba6f2354eb1424cba2fdc4d020cc90ec5817c62523a41acbcc80c1715f2	1585846838	4/2/2020 5:00:38 PM	0x0000000000000000000000000000000000000000	0xaaba235fb01f054f596f62798efb762b899f93c3	11,111	0x30b216dfa8548dbb76d6abac892d8aec35fbf550	IZTECH Token	IZTECH Token
0x95ce84bc8d6311726b400c12064655cc2c5c7cef76ae80e4af1c5a87e401f09a	1585846883	4/2/2020 5:01:23 PM	0xaaba235fb01f054f596f62798efb762b899f93c3	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	4,444	0x30b216dfa8548dbb76d6abac892d8aec35fbf550	IZTECH Token	IZTECH Token
0xd0996ad0bcfe95c2bec44abb5b4c2d8ccdf67f6fcba343621762673d6af0a1e9	1585846913	4/2/2020 5:01:53 PM	0xaaba235fb01f054f596ff62798efb762b899f93c3	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	1,234	0x30b216dfa8548dbb76d6abac892d8aec35fbf550	IZTECH Token	IZTECH Token
0x20dc93d08f582c598d578be564b45d0a1a91814807aa41f6102f98c080aeeb9b	1585847453	4/2/2020 5:10:53 PM	0x0000000000000000000000000000000000000000	0xaaba235fb01f054f596f62798efb762b899f93c3	100,000	0xcc2205a404e2074f02f1d135e93fff0a7c7fa48d	IZTECH Token	IZTECH Token

Table 4.2.(Cont'd) All Token Transactions on a Wallet

Txhash	UnixTimestamp	DateTime	From	To	Value	ContractAddress	TokenName	TokenSymbol
0x1eb67d122f3aec58c1ae5e1e1202600fc636be2fe783e73851889f81b5879cf5	1585847498	4/2/2020 5:11:38 PM	0xaaba235fb01f054f596f62798efb762b899f93c3	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	2,222	0xcc2205a404e2074f02f1d135e93fff0a7c7fa48d	IZTECH Token	IZTECH Token
0x2f957d422eb027d9a033c01fd71fca7a4ed5942f4882dfdf15d5e2ce7e090d39	1585848608	4/2/2020 5:30:08 PM	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	0xaaba235fb01f054f596f62798efb762b899f93c3	222	0x2bd0c89322ec655eaa98996e0133096b25694bd4	IZTECH Token	IZTECH Token
0xbd1a77fe206a80452f9777f5a15276432e05a28ed9bc8fe572e0e54ea286b557	1585848668	4/2/2020 5:31:08 PM	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	0xaaba235fb01f054f596f62798efb762b899f93c3	222	0x2bd0c89322ec655eaa98996e0133096b25694bd4	IZTECH Token	IZTECH Token
0x427d89b350cdd547b69e9fede759185957ac24684f8516c5a1bb5fb43e647092	1585849448	4/2/2020 5:44:08 PM	0x000000000000000000000000000000000000000	0xaaba235fb01f054f596f62798efb762b899f93c3	100,000	0xc89790a04531da12c455d3f03aeb33e309647e19	IZTECH Token	IZTECH Token
0x3a47b5a6be2200e1c3acb2841f46f45b29451fd722eee655b5576bbdec468b2e	1585849793	4/2/2020 5:49:53 PM	0x000000000000000000000000000000000000000	0xaaba235fb01f054f596f62798efb762b899f93c3	100,000	0x2a971930703e427cf64f4850b2f656f2e67fd014	IZTECH Token	IZTECH Token
0x762a0639b3a87a50ad952a8c3ba00486880add76e7e803864a98f49fbb09dd51	1585849898	4/2/2020 5:51:38 PM	0xaaba235fb01f054f596f62798efb762b899f93c3	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	222	0x2a971930703e427cf64f4850b2f656f2e67fd014	IZTECH Token	IZTECH Token
0x1683cd485cfa13c25b9e74ea3c5d71607480d08b6add7487cb296cbdd72c15a6	1585849958	4/2/2020 5:52:38 PM	0xaaba235fb01f054f596f62798efb762b899f93c3	0x3cb2fac5351fe7dd5bb4b4e4ad8775cc2f312e2c	222	0x2a971930703e427cf64f4850b2f656f2e67fd014	IZTECH Token	IZTECH Token
0x6f6873623a5fcb08d01a2e2e16d808db913b6c1d0f6d1daa68852c4fb26c2c28	1585850078	4/2/2020 5:54:38 PM	0xaaba235fb01f054f596f62798efb762b899f93c3	0x6c4f653f8454d856f737c2cb38cef7e1ad1c8e00	1,234	0x2a971930703e427cf64f4850b2f656f2e67fd014	IZTECH Token	IZTECH Token
0xa1ee9439ed1070a9cd04e8f04ae0775debe083a98bb19f3c0c127eba68e2fffb	1585860553	4/2/2020 8:49:13 PM	0xaaba235fb01f054f596f62798efb762b899f93c3	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	5,321	0x2a971930703e427cf64f4850b2f656f2e67fd014	IZTECH Token	IZTECH Token
0x95263a04eb2569d4df1cbef31ad87103da4bf39015a287323e30d210db8ec542	1585860568	4/2/2020 8:49:28 PM	0xaaba235fb01f054f596f62798efb762b899f93c3	0xd650ccb5ee09d52d5dfcc27ea3437551b64eb59a	5,321	0x2a971930703e427cf64f4850b2f656f2e67fd014	IZTECH Token	IZTECH Token

The amount of token the stakeholders have is shown on the Figure 4.6 below. It can be clearly seen on the pie chart that the manufacturer in the designed system, which is 0xaaba235fb01f054f596f62798efb762b899f93c3, occupies the biggest part of the chart.

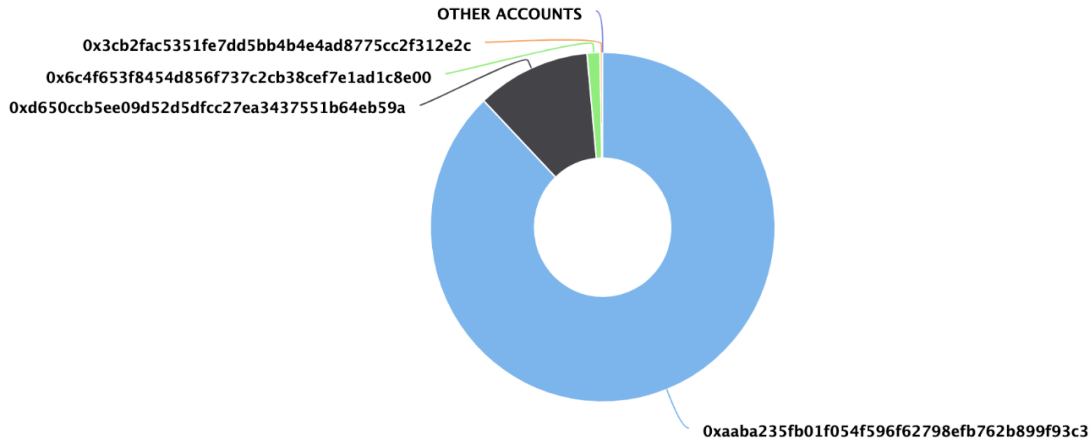

Figure 4.6. IZTECH Token Top 100 Token Holders

The screenshot of MetaMask in Figure 4.7 indicates the confirmation steps of the transaction. The gas fee amount and how many tokens will be transferred can be seen below.

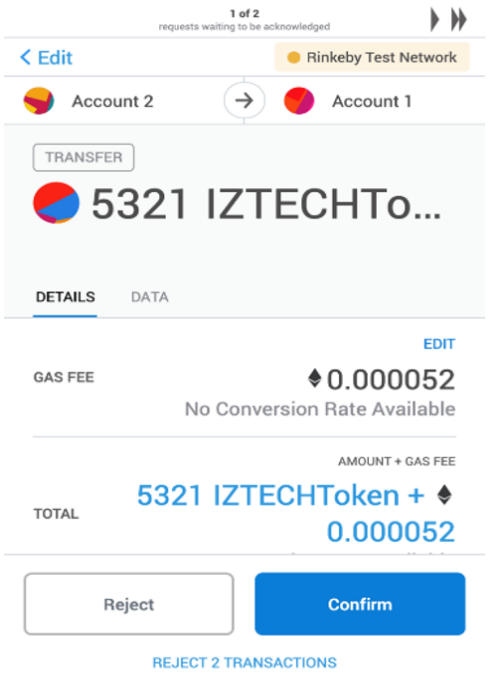

Figure 4.7. Transaction Confirmation Steps in MetaMask

CHAPTER 5

DISCUSSION AND CONCLUSION

Today, there are many types of research that investigate the tokens on Blockchain and are used in different fields. In addition, articles that are similar to the new generation loyalty systems proposed in this thesis can be found. The loyalty system in these articles are briefly as follows: the customer owning 10 stamps on the card earns $ 5 ether from the system; the award that works on the neo chain can be won by reading the QR code on the phone. Additionally, the reward systems of the investigated research studies have been created without complying with a standardized token library and the established system has the outdated Solidity version. On the other hand, new generation loyalty systems are used by some private companies, but companies keep the structural and technological information of their smart contracts confidential due to their commercial concerns.

With the rapid development of technology, it is observed that the systems that cannot catch the technology are increasingly becoming less attractive by the customers. In this thesis, traditional loyalty programs are reinterpreted from a Blockchain perspective. The installed loyalty system works on the Ethereum network and the tokens produced according to ERC20 standards, which are named as IZTECH Token, are given to customers as an award.

The installed system works on Ethereum chain and smart contracts have been created with Solidity language. The relationship between Ethereum nodes and the user is provided by the frontend, which is written using React.js and connected to the Ethereum network with Web3.js. The local Blockchain was created with Ganache and was used during the development phase and after the completion of the developments, the smart contract was migrated to the Rinkeby network.

The system was designed in order to gain an advantage to the manufacturer, market, and the customers. Within the system, the manufacturer becomes the owner of IZTECH Token contract, and distributes the tokens to the markets, which are contracted

with them. When customers purchase products or services, markets give IZTECH Tokens to them along with products and services. This feature allows users to spend the rewards they earn according to their interests. In this way, the user can use the accumulated tokens for their next purchase or can buy ether according to the equivalent of the ether in the stock exchange. In terms of markets, they can buy ether from the manufacturer according to the number of tokens accumulated. Since customers in the system tend to shop more to earn more tokens, thus markets sell more products and consequently manufacturers produce more products. As a result, it was aimed to make profit for the manufacturer, market, and customers in the established loyalty system and a tiny economy was established.

5.1. Future Work

The loyalty system can be followed over the Rinkeby network and can be seen by everyone. However, since it is costly to migrate and run the smart contract, the loyalty system mentioned in this thesis is currently not available to the main net. The proposed loyalty system can be moved to the main net in the future if sponsored by companies. In order to increase the value of IZTECH Tokens in the stock market, statistical studies can be conducted on the stock exchange token standards and token amounts. Due to the smart contract structure was developed in a modular structure, new features can be added in line with the requests received by companies and the structure can be enlarged.

REFERENCES

[1] Zhang, Y., & Wen, J., An IoT Electric Business Model Based on The Protocol of Bitcoin, 18th International Conference on Intelligence in Next Generation Networks, IEEE, (2015), 184-191.

[2] Sharples, M., & Domingue, J., The Blockchain And Kudos: A Distributed System for Educational Record, Reputation and Reward, In European Conference on Technology Enhanced Learning, Springer, Cham., (2016), 490-496.

[3] Ayed, A. B., A Conceptual Secure Blockchain-Based Electronic Voting System, International Journal of Network Security & Its Applications, 9 (3), (2017), 01-09.

[4] The-cma.org. (2020). The 2017 COLLOQUY Loyalty Census. [online] Available at: https://www.the-cma.org/Contents/Item/Display/327325 [Accessed 15 Feb. 2020].

[5] Shelper, P., Lowe, A., & Kanhere, S. S., Experiences from the Field: Unify Rewards-A Cryptocurrency Loyalty Program, In Proceedings of the Symposium on Foundations and Applications of Blockchain, (2018).

[6] GitHub. (2019). OpenZeppelin/openzeppelin-contracts. [online] Available at: https://github.com/OpenZeppelin/openzeppelin-contracts/tree/master/contracts [Accessed 13 Dec. 2019].

[7] Underwood, S., Blockchain Beyond Bitcoin, (2016), 15-17.

[8] Nakamoto, S., Bitcoin: A Peer-To-Peer Electronic Cash System, Bitcoin, (2008).

[9] CoinSutra - Bitcoin Community. (2020). What is Double Spending & How Does Bitcoin Handle It? [online] Available at: https://coinsutra.com/bitcoin-double-spending/ [Accessed 4 Feb. 2020].

[10] Nakamoto, S., Bitcoin: A Peer-to-Peer Electronic Cash System, Manubot, (2019).

[11] Galbraith, S. D., & Gaudry, P., Recent Progress on the Elliptic Curve Discrete Logarithm Problem, Designs, Codes and Cryptography, 78(1), (2016), 51-72.

[12] Diffie, W., & Hellman, M., New Directions in Cryptography, IEEE Transactions on Information Theory, 22(6), (1976), 644-654.

[13] Merkle, R. C., A Certified Digital Signature. In Conference on the Theory and Application of Cryptology, Springer, New York, NY, (1989), 218-238.

[14] Zheng, Z., Xie, S., Dai, H., Chen, X., & Wang, H., An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends, In 2017 IEEE International Congress on Big Data (BigData Congress), IEEE, (2017), 557-564.

[15] Di Pierro, M., What is the Blockchain? Computing in Science & Engineering, 19(5), (2017), 92-95.

[16] Haffke, F., Technical Analysis of Established Blockchain Systems, Technical University of Munich, SW Engineering for Business Informatics, (2017).

[17] Bitcoin.org. (2020). Blockchain Guide - Bitcoin. [online] Available at: https://bitcoin.org/en/blockchain-guide#proof-of-work [Accessed 4 Feb. 2020].

[18] GitHub. (2020). Ethereum. [online] Available at: https://github.com/ethereum/wiki/wiki/white-paper#infrastructure [Accessed 4 Feb. 2020].

[19] GeeksforGeeks. (2020). Practical Byzantine Fault Tolerance (pBFT) - [online] Available at: https://www.geeksforgeeks.org/practical-byzantine-fault-tolerancepbft/ [Accessed 4 Feb. 2020].

[20] Crush Crypto. (2020). What is Delegated Proof of Stake? - Crush Crypto. [online] Available at: https://crushcrypto.com/what-is-delegated-proof-of-stake/ [Accessed 4 Feb. 2020].

[21] Rouhani, S., & Deters, R., Performance Analysis of Ethereum Transactions in Private Blockchain. In 2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS), IEEE, (2017), 70-74.

[22] Stateofthedapps.com. (2020). State of the DApps — DApp Statistics. [online] Available at: https://www.stateofthedapps.com/stats [Accessed 4 Feb. 2020].

[23] Coin Market Cap. (2020). Crypto Money Market Value. [online] Available at: https://coinmarketcap.com/tr/ [Accessed 4 Feb. 2020].

[24] Laurence, T., Blockchain for Dummies, John Wiley & Sons, (2019).

[25] Remix.ethereum.org. (2020). Remix - Ethereum IDE. [online] Available at: https://remix.ethereum.org/ [Accessed 4 Feb. 2020].

[26] Berman, B., Developing an Effective Customer Loyalty Program, California Management Review, 49(1), (2006), 123-148.

[27] Agrawal, D., Natalia, N., Gopalakrishnan, G., Guzman, M. N., McDonald, M., & Kim, H. M., Loyalty Points on the Blockchain, Business and Management Studies, 4(3), (2018), 80-92.

[28] Wang, L., Luo, X. R., & Xue, B., Too Good to Be True? Understanding How Blockchain Revolutionizes Loyalty Programs, (2018).

[29] Avital, M., Peer review: Toward a Blockchain-Enabled Market-Based Ecosystem, Communications of the Association for Information Systems, 42(1), (2018), 28.

[30] Díaz, M., Massó, J., Pérez-Villegas, A., & Sagalés, A. Smart Token for The Campus Using Blockchain, (2017).

[31] Liao, C. H., Teng, Y. W., & Yuan, S. M., Blockchain-Based Cross-Organizational Integrated Platform for Issuing and Redeeming Reward Points, In Proceedings of the Tenth International Symposium on Information and Communication Technology, (2019), 407-411.

[32] Bülbül, Ş., & İnce, G, Blockchain-based Framework for Customer Loyalty Program, 3rd International Conference on Computer Science and Engineering (UBMK), IEEE, (2018), 342-346.

[33] Docs.openzeppelin.com. (2020). OpenZeppelin Documentation - OpenZeppelin Docs. [online] Available at: https://docs.openzeppelin.com/openzeppelin/ [Accessed 14 Feb. 2020].

[34] Anonymous, (2020). [online] Available at: https://rinkeby.etherscan.io/ [Accessed 13 Feb. 2020].

[35] Faucet.rinkeby.io. (2020). Rinkeby: Authenticated Faucet. [online] Available at: https://faucet.rinkeby.io/ [Accessed 13 Feb. 2020].

[36] Truffle Suite. (2019). Truffle | Overview | Documentation | Truffle Suite. [online] Available at: https://www.trufflesuite.com/docs/truffle/overview [Accessed 13 Dec. 2019].

[37] Truffle Suite. (2019). Ganache | Truffle Suite. [online] Available at: https://www.trufflesuite.com/ganache [Accessed 13 Dec. 2019].

[38] Tr.reactjs.org. (2019). – React. [online] Available at: https://tr.reactjs.org/docs/getting-started.html [Accessed 13 Dec. 2019].

[39] MetaMask.io. (2019). MetaMask. [online] Available at: https://metamask.io/ [Accessed 14 Dec. 2019].

[40] Infura. (2020). Ethereum API | IPFS API & Gateway | ETH Nodes as a Service | Infura. [online] Available at: https://infura.io/ [Accessed 10 Feb. 2020].

[41] Lee, D. R., Jang, Y., & Kim, H., Poster: A Proof-of-Stake (PoS) Blockchain Protocol using Fair and Dynamic Sharding Management, In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, ACM, (2019), 2553-2555.

[42] Solidity.readthedocs.io. (2020). Solidity — Solidity 0.5.0 Documentation. [online] Available at: https://solidity.readthedocs.io/en/v0.5.0/ [Accessed 10 Feb. 2020].

[43] Rinkeby.Etherscan.io. (2020). IZTECHToken. [online] Available at: https://rinkeby.etherscan.io/token/0x2a971930703e427cf64f4850b2f656f2e67fd014 [Accessed 11 April 2020].

[44] Rinkeby.Etherscan.io. (2020). Address 0xaaba235fB01f054F596f62798efB762B899F93C3. [online] Available at: https://rinkeby.etherscan.io/address/0xaaba235fb01f054f596f62798efb762b899f93c3#tokentxns [Accessed 11 April 2020].

APPENDIX

SMART CONTRACT OF IZTECH TOKEN

```
contract IZTECHToken is ERC20Mintable {

    using SafeMath for uint256;
    string public name = "IZTECHToken";
    string public symbol = "IZTECHToken";
    uint256 public decimals = 0;
    mapping(address => bool) awarders;
    //list of award "types" mapped to the awarder
    mapping(uint => address) awards;
    address private _owner;
    event AwardGiven(address indexed _from, address indexed _to,  uint indexed _type,
uint _amount, uint _date);
    event AwardRevoked(address indexed _from, address indexed _to,  uint indexed
_type, uint _date);
    event AwardAdded(address _from, uint _type);
    event AwarderAdded(address _who);
    event AwarderRemoved(address _who);

    constructor () public {
        _owner = _msgSender();
        mint(_msgSender(),100000);
    }

    function giveAward(address user, uint awardId, uint256 amount, uint date, uint256
rate)  public {

        require (awards[awardId] != msg.sender,'The awarder has to own the award in
order to award it' );
```

```
        uint256 tokenAmount = (amount).div(rate);
        require (super.balanceOf(msg.sender) > tokenAmount, 'the awarder has to have
enough supply to give the points');
        super.transfer(user, tokenAmount);
        if (date == 0) {
            date = now;
        }
        emit AwardGiven(_msgSender(), user, awardId, amount, date);

    }

    function revokeAward(address user, uint awardId, uint date) public {

        require(awards[awardId] != msg.sender);
        if (date == 0) {
            date = now;
        }
        emit AwardRevoked(_msgSender(), user, awardId, date);

    }

    function isAwarder(address _addr) view public returns (bool) {
        return awarders[_addr];
    }

    modifier onlyOwner() {
        require(_msgSender() == _owner,"you should be owner");
        _;
    }
    function deneme(address payable account, uint256 amount) payable public returns
(bool) {
        account.transfer(amount.mul(1000000000000000000));
        return true;
    }
```

```
    modifier onlyAwarder(address _addr) {
      require(awarders[_addr]);
      _;
    }

    function addAwarder(address _addr) public onlyOwner {
      awarders[_addr] = true;
      emit AwarderAdded(_addr);
    }

    function deleteAwarder(address _addr) public onlyOwner {
      awarders[_addr] = false;
      emit AwarderRemoved(_addr);
    }

    function addAward(uint index) public returns (bool) {
      require (!isAwarder(_msgSender()));

      if (awards[index] == address(0)) {
        awards[index] = _msgSender();
        emit AwardAdded(_msgSender(), index);
        return true;
      }
      return false;

    }

    function getAward(uint index) view public returns (address) {
      return awards[index];
    }

    function payBack(uint256 amount, address payable retail, uint256 exchange ) public
payable onlyAwarder(retail) onlyOwner returns(bool) {
```

```
uint256 payBackValue = (amount.div(exchange)).mul(1000000000000000000);
        require ( _msgSender().balance > payBackValue,"Doesn't have enough ether.");
        retail.transfer(payBackValue);
        destroyToken(retail, amount);
        return true;
    }
}
```