Mitigating the Financial Risk of Ethereum Based Lending System

This thesis/project is submitted in partial fulfillment of the requirement for the degree of Bachelor of Science in Computer Science & Engineering.

Saha Reno

ID: 1101086

Md. Yeasin Arafat

ID: 1101017

Supervised by

Mamun Ahmed

Assistant Professor

Department of Computer Science & Engineering (CSE)

Bangladesh Army International University of Science & Technology (BAIUST)

Department of Computer Science & Engineering

Bangladesh Army International University of Science & Technology (BAIUST)

Cumilla Cantonment, Cumilla.

Acknowledgement

First and foremost, I would like to thank God Almighty for giving me the strength to finish this work. The satisfaction that accompanies the successful completion of this thesis would be incomplete without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success. I am grateful to my honorable project Supervisor Mamun Ahmed, Assistant Professor, Department of Computer Science and Engineering, Bangladesh Army International University of Science & Technology (BAIUST), for the guidance, inspiration and constructive suggestions which were helpful in the preparation of this project. I also convey special thanks and gratitude to Jahidul Arafat, honorable head of the Department of Computer Science and Engineering, Bangladesh Army International University of Science & Technology (BAIUST), for his kind advice. I would also like to extend my gratitude to all of my teachers for their valuable guidance in every step of mine during the four years of learning stage. Finally I would like to thank my friends, senior, juniors and the staffs of the department for their valuable suggestion and assistance that has helped in successful completion of the project.

Abstract

Lending systems in real world are not that much secure as the lenders may face financial loss because of different types of fraud involved in lending. The borrower may run away with the money he borrowed from the lender and the possibility of removing all the traces of the transactional history makes the loan system much vulnerable. Any third party involved in this aspect may also create deceitful situations (such as: destructing the documents of evidence where the borrower signed an agreement). BlockChain is a secure system where the transactional history regarding crypto-currency can not be modified or destructed. Ethereum is a protocol which is based on BlockChain technology has several benefits over other crypto-currency based systems and is best suited for creating a secure lending system. Every Ethereum based system runs on 'Smart-Contracts' which are lines of code and makes the system automated. As the system gets automated, proper algorithms can make the system reliable and secure as each and every steps of the system are maintained and executed by the algorithm inside the Smart-Contracts. Because of its reliability and security, a secure lending system can be made using the Ethereum protocol where the smart-contract will be written in such a way that the borrowed amount must get returned to the lender after a certain days. Some Ethereum based lending systems are available in Ethereum network but these systems have several deficiencies. One of the major deficiencies of these systems is the transactional unit is set directly in Ether. If the exchange rate of Ether in dollar decreases at the time of returning the borrowed money, the lenders has to face financial loss. Another decrement of the system is that ERC-20 Token's value, which is used as collateral, is fixed on Ether and not on paper-money. If someone buys some ERC-20 Tokens for taking loans in the near future and the exchange rate of Ethers in dollars falls down, then the person has to face loss of money. Moreover, if these collateral's value can be incremented periodically by a certain amount, then more people will be interested to join the system to buy tokens as it will financially benefit them. The proposed system mitigates the business risk by removing the deficiency mentioned above. This system fixes the transactional unit in paper-money and increments the ERC-20 token's value periodically upto a certain amount.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Overview

A blockchain is a public ledger of all bitcoin transactions that have ever been executed. The records of transactions made in Bitcoin or another cryptocurrency are stored in blocks and maintained across all the computers that are linked in a peer-to-peer network. A block is the "current" part of a blockchain which records some or all of the recent transactions, and once completed, goes into the blockchain as permanent database. Each time a block gets completed, a new block is generated. Blocks are linked to each other (like a chain) in proper linear, chronological order with every block containing a hash of the previous block. To use conventional banking as an analogy, the blockchain is like a full history of banking transactions. Bitcoin transactions are entered chronologically in a blockchain just the way bank transactions are. Meanwhile, blocks, are like individual bank statements. The full copy of the blockchain has records of every bitcoin transaction ever executed. It can thus provide insight about facts like how much value belonged to a particular address at any point in the past. Some developers have begun looking at the creation of other different blockchains as they do not believe on depending on a single blockchain. Parallel blockchains and sidechains allow for tradeoffs and improved scalability using alternative, completely independent blockchains, thus, allowing for more innovation. It secures the transactions in a way that any record of transaction that occurred in the past cannot be modified as the modification changes the hash of several blocks. Changing the hash of the blocks will inevitably result in the breakage of the links among the blocks. Also all the peers connected to that system does not support that modification excluding the modifier, which is based on consensus algorithm.

Like Bitcoin, Ethereum is a distributed public blockchain network. It is a cryptocurrency based system that is built on the Blockchain technology where the cryptocurrency is called 'Ether'.There is a type of token that is used to pay miners fees for including transactions in their block, it is called gas, and every smart contract execution requires a certain amount of gas to be sent along with it to entice miners to put it in the blockchain. Although there are some significant technical differences between Bitcoin and Ethereum, the most important distinction to note is that they differ substantially in purpose and capability. Bitcoin offers one particular application of blockchain technology, a peer to peer electronic cash system that enables online Bitcoin payments. While the Bitcoin blockchain is used to track ownership of digital currency (bitcoins), the Ethereum blockchain focuses on running the programming code of any decentralized application. Ethereum has got many facilities over other Blockchain based cryptocurrency systems. The mining process of almost every cryptocurrency system consumes huge energy which is not bearable at all for most of the countries in the world. Moreover, most of these systems are not open-source. Ethereum removes all these deficiencies, is open-source where developers from all around the world can use this system in different aspects and therefore, is preferable over all other Blockchain based systems. Like other BlockChain based systems, this Ethereum protocol can be used to secure any virtual transactions as the blocks containing the transactions records are not alterable at any means. Many organizations all over the world are now using Ethereum based services to ensure anonymity, accuracy, efficiency and most importantly, security.

1.2 Background

Lending system in physical world is not secured as the person being loaned may run away with the money he borrowed from the lender. Any third party involved in this aspect may also create deceitful situations (such as: destructing the documents of evidence where the borrower signed an agreement) which causes the lender to face financial loss. The upshot of the above discussion is that the possibility of removing all the traces of the loan-transaction makes the loan system much vulnerable. As it is impossible to modify or destroy any record of transactions from a block of Blockchain, the loan system can be made secured using the methodology of Blockchain technology.

Ethereum is considered as the best choice for implementing such lending systems as mining a transaction requires a very small amount of time compared to other BlockChain based cryp-

tocurrency systems. Also it provides users the facility of creating their own cryptocurrency besides Ether inside the system which is formally known as Tokens. These tokens can be reliably and efficiently used as the Digital Asset for Ethereum based lending systems and are substitutes for the Physical Assets which is required to take loans from organizations like bank etc. in the real world to make lending systems more secure.

ERC-20 is a technical standard, widely utilized for the tokens used as Digital Assets in almost all Ethereum based lending systems. This standard is followed due to following reasons:

1. For starters, once a smart contract is deployed, it cannot be changed anymore, so if you made an error you can't fix it. That could be quite catastrophic. Imagine a bug inside your contract's code that causes people to lose their tokens or a bug that would allow others to steal tokens.

2. And then there is the problem of interoperability. Each token contract can be completely different from the other. So if you want your token to be available on an exchange, the exchange has to write custom code so they can talk to your contract and allow people to trade.

3. Same thing goes for wallet providers. Supporting hundreds of tokens would be very complex and time consuming.

So instead, the community proposed a standard called ERC20. This ERC20 is a guideline or standard for when you want to create your own token.

1.3 Research Problem

Many protocols have been created to implement the lending system based on Ethereum. Some of these systems provide collateralized loans, while others provide uncollateralized loans. Systems which provide collateralized loans mostly use ERC-20 tokens as collateral. Although, it is the best alternative of physical assets in BlockChain based loan systems, there are mainly two issues regarding the use of ERC-20 standard tokens inside these lending systems:

1. The basic ERC-20 standard fixes the market value of these tokens on Ether and not on physical currencies (e.g. USD, EUR etc.). Thus the price of this token varies from time to time as the price of Ether fluctuates over time. If, at any time, the rate of Ether falls down, the exchange rate in dollars for these tokens will also go downward. People who

bought tokens and stored them in their wallets for future use and the lenders who lent money in exchange of these tokens will then have to face loss of money and will therefore be discouraged to lend his money from the next time.

2. ERC-20 standard tokens has no pre-installed beneficial characteristics rather than getting the facility of borrowing money which will encourage the system users to buy these tokens and keep them in their virtual wallet. It brings no benefit to the users in other ways.

Moreover, putting Ether as the lending material may bring loss to the lenders. If, the exchange rate of the Ether in dollars may fall at the time of borrowers returning the Ethers he/she borrowed from the lenders, the lender will eventually face financial loss. Making the lending system based on paper money and not directly on Ether solves this problem.

1.4 Research Objectives

If the deficiencies mentioned above can be resolved, an efficient lending system can be constructed which will benefit the lenders economically. These weaknesses of existing Ethereum based lending systems is sorted out by the proposed system in the following means:

1. The implemented protocol fixes the exchange rate of ERC-20 tokens mainly on physical currency and not directly on Ether. Therefore, even if the price of Ether falls down, the buyer has to pay extra Ethers for each of the tokens as the extra amount is needed to match the exchange rate in USD.

2. The system increases the market value of this Digital Asset periodically by a little amount. This way, more people will be encouraged to buy tokens and store them in their wallets as this process will financially benefit the token holders.

3. Unlike other lending protocols, where the task of borrowing and lending is done by Ether, the transactional unit of the system is paper money (USD). The system converts the paper money value into equivalent ethers and at the time of returning money to the lenders, the system will check if the fluctuation in the exchange rate of ethers in dollars causes the rise of the amount of Ethers to be returned. If the exchange rate of Ether in dollar falls down, then the system will update the amount of ethers to be returned and will bring out the additional amount of ethers from the borrower's account.

1.5 Scope

The proposed thesis implements a system which guarantees no financial loss to the lender by only supporting collateralized loans. ERC-20 token is one of the most popular Ethereum tokens and hence, it is chosen as the collateral for loans. Borrowers are required to provide adequate ERC-20 tokens to take loans from the lenders. This system does not support any uncollateralized loans, where the risk assessment will be calculated by an efficient algorithm which will analyse all the previous transactional history of the borrower's account and will decide whether it will be safe to provide loan to that borrower. Moreover, the system currently does not support other than ERC-20 tokens, where there are many more popular Ethereum tokens like ERC-223, ERC-777, Binance Coin, Maker, OmiseGo, Basic Attention Token etc.

1.6 Thesis outline & organization

The remainder of the thesis is organized as follows. Literature Review chapter describes the related terminologies and existing works related to the thesis. In the next chapter named System Overview, the working procedure and the efficiency of the proposed system are described. Implementation Details chapter focuses on how the system is implemented using various programming languages, frameworks and tools related to the BlockChain technology. In Experimental Results & Discussion chapter, the output produced by the system is described and demonstrated using images at different stages of a lending. The final chapter, which is Conclusion and Future Recommendation, summarizes the whole aspect of the implemented system.

Chapter 2

Literature Review

2.1 Overview

2.1.1 Bitcoin: A Peer-to-Peer Electronic Cash System[1]

Satoshi Nakamoto proposed the idea of a decentralized, peer-to-peer electronic cash system, named Bitcoin in 2008. This Bitcoin creates ledger of transactions.

Identification: each user has their own unique address, which is basically the hash of their own public key. Private key of the corresponding public key is used to sign the transaction. Bitcoins are sent from one address to another.

Transaction: A user has to create two proofs to send Bitcoins to another user. At first, the user who is sending Bitcoins must show where he had acquired the funds from that he is sending. Transaction is the only way to receive funds. so the user provides a pointer to the transaction via which he had received the funds. Secondly, the user must prove that he/she has control over the source of funds which he/she is trying to transfer to another account. Funds are kept in addresses which corresponds to specific public keys. The user must use the private key to provide a digital signature to the public key to authorize the transaction. The user signs the entire transaction so the internal contents of that transaction cannot be modified. After successful creation of the transaction, it is broadcast to each peer in the network and added to ledger. Figure 2.1 demonstrates the above concept.

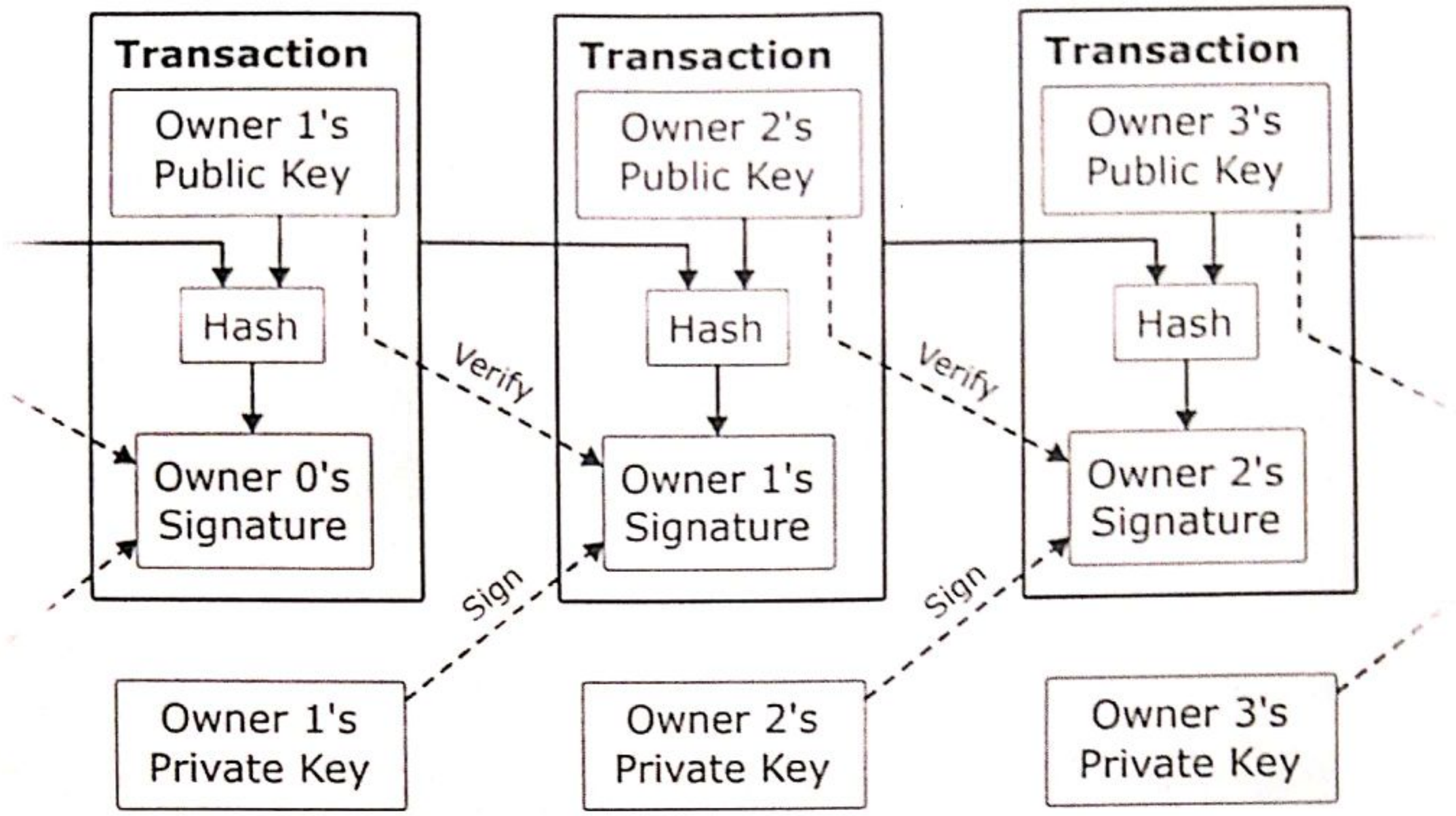

Figure 2.1: Bitcoin Transaction Method Using BlockChain[16]

Double Spending: The Bitcoin transaction scheme prevents the fraud which occurs in coin funding: however, it does not provide any protection against double-spending. Suppose Reno sends a coin to Pranto; later on, he sends the same coin to Rajon as well. Broadcasting both transactions using the same coin would create confusion as to which transaction is legitimate. Bitcoin uses proof-of-work as a consensus protocol to prevent double spending.

Proof-of-work: Blockchain network consists of several blocks chained with each other, each of which are groups of transactions bundled together. To prevent double spending or a malicious attack to modify any information stored in the blocks, each block comes with a proof-of-work, performed by a miner. A block is only accepted if the hash of the block satisfies certain restraints. Each block has a nonce (see Figure 2.2) that is modified repeatedly in order to validate transactions. This process is known as mining. The creator of a block is allowed to add one additional transaction that stores a small amount of coins in their account. Whenever a miner successfully mines a block, the block is added to the blockchain. Each block references the pre-

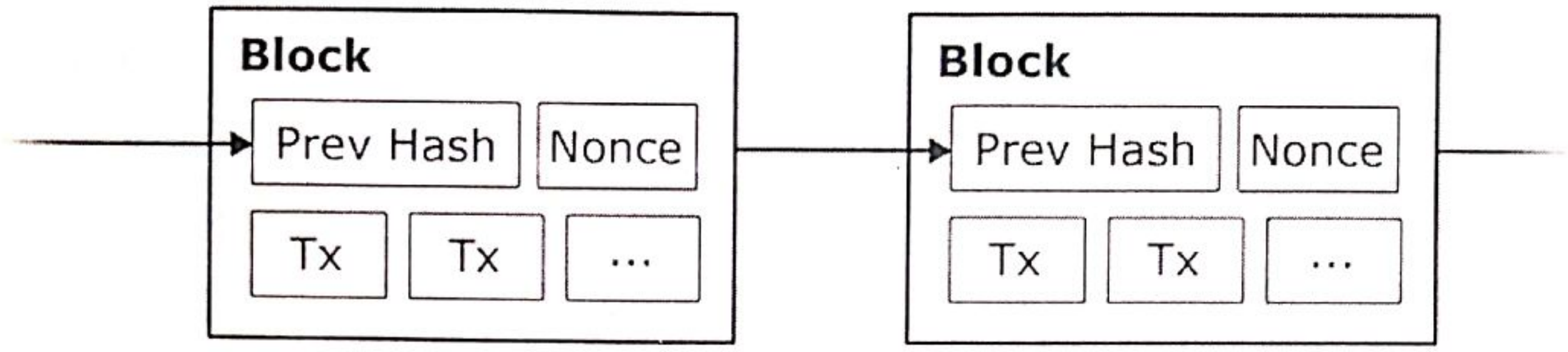

Figure 2.2: Proof of Work in Bitcoin Transactions[16]

vious block of the chain. At any moment, there may exists multiple chains of blocks branching from each other. To create consensus, miners only mine upon the largest chain and discard the smaller ones.

2.1.2 Ethereum[5]

Bitcoin provides a simple stack based programming language known as Script, which can be used to create features such as multi-signatures and escrow transactions. Although this Script can be considered to be a programming language, it is not Turing complete and its capabilities are limited. The main reason behind this is that a Turing complete blockchain language can be easily taken advantage of. A simple infinite loop can create great complications in detecting which computations are needed to be performed. As an alternative to the Bitcoin system, Ethereum provides Turing complete smart contracts that run on the EVM.

Ethereum Details: Ethereum is considered to be a state-transition system. The Ethereum system is composed of objects known as accounts. There are two main types of accounts: "externally owned accounts" and "contract accounts." Externally owned accounts are controlled by private keys while contract accounts are fully autonomous and governed by their contract code. Both types of accounts have the ability to send messages and create new accounts. A message is essentially a transaction. Messages transfer Ethers, which is the currency for Ethereum, from account to account. If a contract account receives a message, its code will be triggered in order to determine what actions to be executed next.

Ethereum Fees and Gas: Ethereum's solution to the "infinite loop/excessive computation problem" is to charge a small amount of fees for the computation. Each message that is sent specifies an amount of Gas for the message. Each computational step requires some Gas. If the Gas is completely spent before the code is completely executed, the execution stops and all changes are reverted. Otherwise, if all the statements of a specific code-block is successfully executed, any leftover gas is returned to the sender of the message. This prevents infinite loops in any smart contracts, as the amount Gas to be supplied for a transaction is finite, forcing all computation to come to an end eventually. This phenomenon also prevents malicious users from attacking the Ethereum network via excessive computation since the amount of computation is proportional to the amount paid.

With the introduction of decentralized smart contracts, several protocols for decentralized lending have been proposed.

2.1.3 Siacoin[13]

Sia is a cloud storage platform which is decentralized in nature. Rather than renting cloud storage from a centralized platform, Sia users rent storage from each other. User who provides storage is known as a host, agrees to store the client's data. The storage provider periodically provides proof of their continued storage. The storage provider is compensated for each proof they provide, but if they miss a proof, they are penalized. The proofs are publicly verifiable and are stored on the blockchain. Thus, the users need not to verify each proof manually.

2.1.4 Monero[14]

Monero is a cryptocurrency which is based on the CryptoNote protocol. CryptoNote provides users with anonymous payment system using a technology named ring signature that allows users to sign messages on behalf of a group. Ring signatures obfuscate the transactions via mixins. Each output describes money that is waiting to be spent by its owner. Each input to a transaction references a list of previous outputs (the mixins), only one of which is actually being spent. This way the blockchain does not reveal the transaction output that is actually being spent. Additionally, the ring signatures are linkable to prevent an output from being spent twice. Monero provides users with persistent cryptographic identities called addresses. In order to hide the user identity, Monero uses Stealth Addresses to blind the address from blockchain

observers. In 2016, Monero deployed ring-confidential transactions (ringCT) (Noether et al., 2016), which hide the amount of money involved in a transaction. In 2017, Monero updated the ringCT protocol to decrease the memory size of the protocol (Sun et al., 2017).

2.1.5 Tumblebit[15]

Tumblebit is a payment protocol which is used for anonymous payments. This Tumblebit protocol is fully compatible with Bitcoin. The Tumblebit protocol operates through a Tumbler that can be used to make payments which are not linkable. The importance of Tumblebit lies in the fact that the Tumbler is untrusted: the Tumbler can neither steal funds nor deanonymize users.

2.1.6 Ethlend[6]

Ethlend is a decentralized lending system which provides collateralized lending to its users. The protocol uses Ethereum ERC-20 Standard tokens as collaterals. A borrower creates a request for Ethers to borrow which specifies the principle, interest rate, collateral, and other factors. The borrower then sends the specified coins to the borrower's account. A lender can choose to fund the contract, which initiates the loan. When the allotted time to return the amount of borrowed Ether has passed, the smart contract which manages all the loan performs on of two tasks:

1. If the borrower is able to fully repay his/her loan, the loan contract returns the ERC-20 tokens from the lender's account to the borrower.

2. Otherwise, the smart contract initiates the transfer of the tokens from borrower's wallet to the lender.

The protocol also allows the lenders to create his/her own loan contract and specify the principle and interest rate. This features is known as On-demand learning. Borrowers then initiate the loan by sending ERC-20 tokens to the loan contract. The Ethlend whitepaper also discusses the possibility of using the ENS system as well.

2.1.7 Dharma Version 1[16]

The Dharma whitepaper made the first attempt of creating a protocol for decentralized lending. The main difference between Dharma and Ethlend is that the Dharma protocol provides

uncollateralized lending, rather than collateralized lending. The Dharma protocol consists of three main "layers."

Loan Layer: Each loan consists of a central smart contract that controls the main logic of the loan.

Origination Layer: This layer is responsible for making a relation between a borrower and a lender who wants to lend money to that borrower.

Risk-Assessment Layer: A RAA, or Risk-Assessment Attestor, assesses both the authenticity of borrower's identity and the credit risk.

2.1.8 Dharma Version 2[17]

The Dharma version 2 protocol improves upon the previous Dharma protocol. The protocol consists of four main types of actors: debtors, underwriters, relayers, and creditors. A basic overview:

Debtors: Debtors are parties that borrow assets and funds.

Creditors: Creditors are parties that lend assets and funds.

Underwriters: In traditional debt markets, underwriters are entities that collect fees for administering the public issuance of debt and pricing borrower default risk into the asset. In Dharma protocol, this definition is expanded and formalized. An underwriter is a trusted entity that collects market-determined fees for performing the following functions:

1. Originating a debt order from a borrower

2. Determining and negotiating the terms of the debt (i.e. term length, interest, amortization) with the potential debtor Cryptographically committing to the likelihood they ascribe to that debt relationship ending in default

3. Administering the debt order's funding by forwarding it to any number of relayers.

4. Servicing the debt – i.e. doing everything in the underwriter's reasonable power to ensure timely repayment according to the agreed upon terms

5. In the case of defaults or delinquencies, collecting on collateral (if debt is secured) or the individual's assets via legal mechanisms and passing collected proceeds to investors

Relayers: Maintain an "orderbook" of loans that creditors can invest in. Relayers essentially connect the creditors to the loans. They aggregate signed debt order messages and, for an agreed upon fee, host the messages in a centralized order book and provide retail investors with the ability to invest in the requested debt orders by filling the signed debt orders. Dharma Protocol relayers need not hold any agent's tokens – they simply provide a mechanism for creditors to browse through aggregated signed debt order messages, which creditors can use to trustlessly issue themselves debt tokens in exchange for the requested principal via client-side contract interactions. Some of the characteristics of Dharma Protocol Relayers are:

1. Dharma protocol relayers are not hosting a secondary market order book, but rather, an order book containing requests for debts that have yet to be issued

2. Dharma protocol relayers provide creditors with signed debt-specific metadata associated with the debt order messages and their accompanying underwriter so that they can make informed investment decisions about the risk profile of a given debt order.

3. Dharma protocol relayers do not freely allow any anonymous party to publish signed debt orders on to their order book, and use their discretion to only accept signed debt orders from known, trusted underwriters.

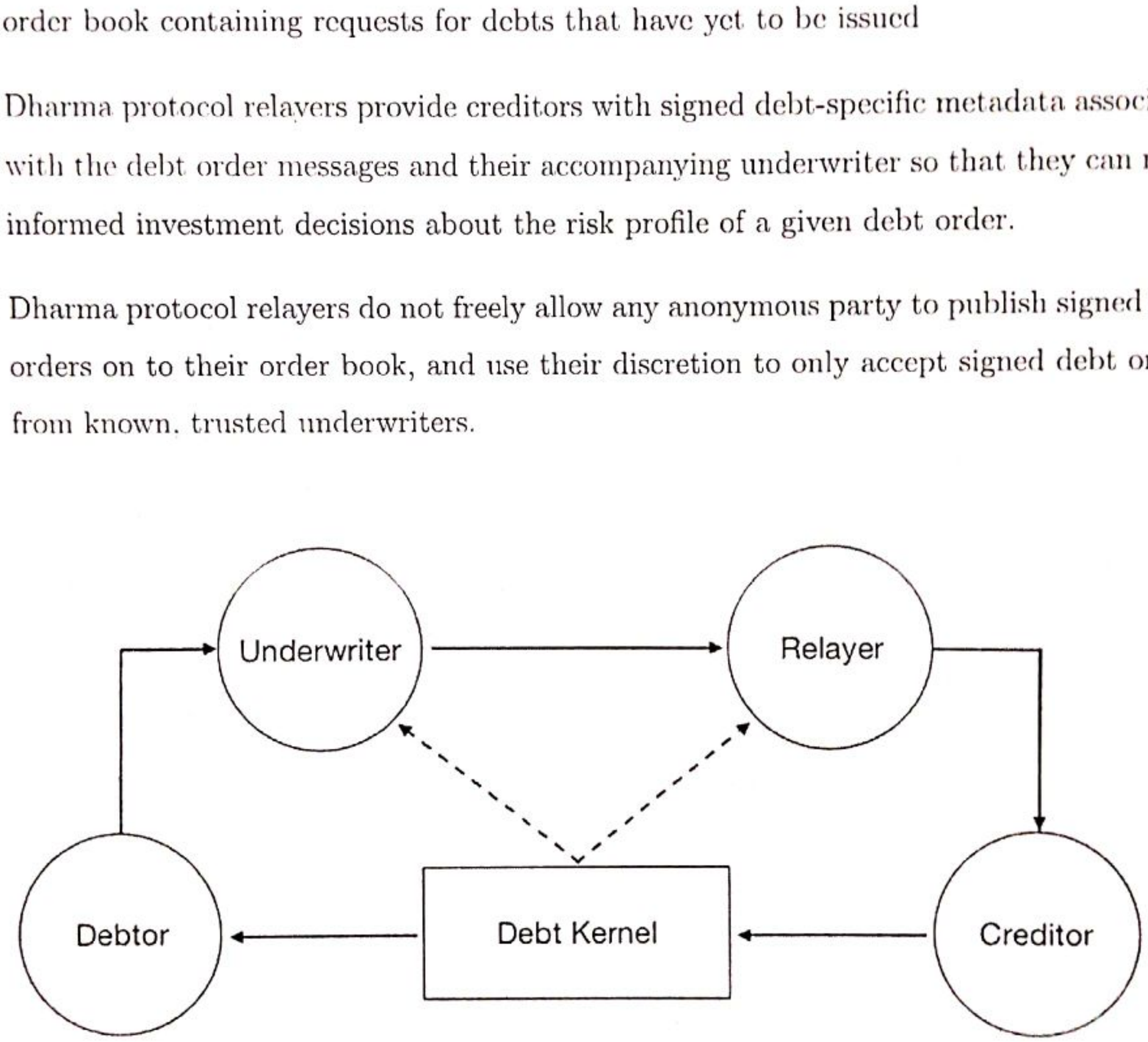

Figure 2.3: Relationship Among Various Entities of Dharma Protocol[17]

Dharma protocol leverages several contracts deployed on the Ethereum network. We highlight a few that are particularly relevant to understanding the prtocol's mechanics.

Debt Kernel: The debt kernel is a simple smart contract that governs all business logic associated with minting non-fungible debt tokens, maintaining mappings between debt tokens and their associated term contracts, routing repayments from debtors to creditors, and routing fees to underwriters and relayers. These mechanisms are easier to define within the context of the debt lifecycle, and are extensively elaborated on in the below specification.

Terms Contract(s): Terms contracts are Ethereum smart contracts that are the means by which debtors and creditors agree upon a common, deterministically defined set of repayment terms. By extension, terms contracts expose a standard interface of methods for both registering debtor repayments, and programmatically querying the repayment status of the debt asset during and after the loan's term. A single terms contract can be reused for any number of debt agreements that adhere to its repayment terms – for instance, a terms contract defining a simple compounded interest repayment scheme can be committed to by any number of debtors and creditors. The exact interface for this is defined within the specification below.

Repayment Router: The repayment router contract is constructed to trustlessly route repayments from debtors to debt agreement beneficiaries (i.e. owners of the debt tokens). Additionally, the repayment router acts as a trusted oracle to the Terms Contract associated with any given debt agreement, reporting to it the exact details of each repayment as it occurs. This enables the terms contract to serve as a trustless interface for determining the default status of a debt

Debtor-Filler Order Submissions: The arrangements in which a debtor fills a complete debt order

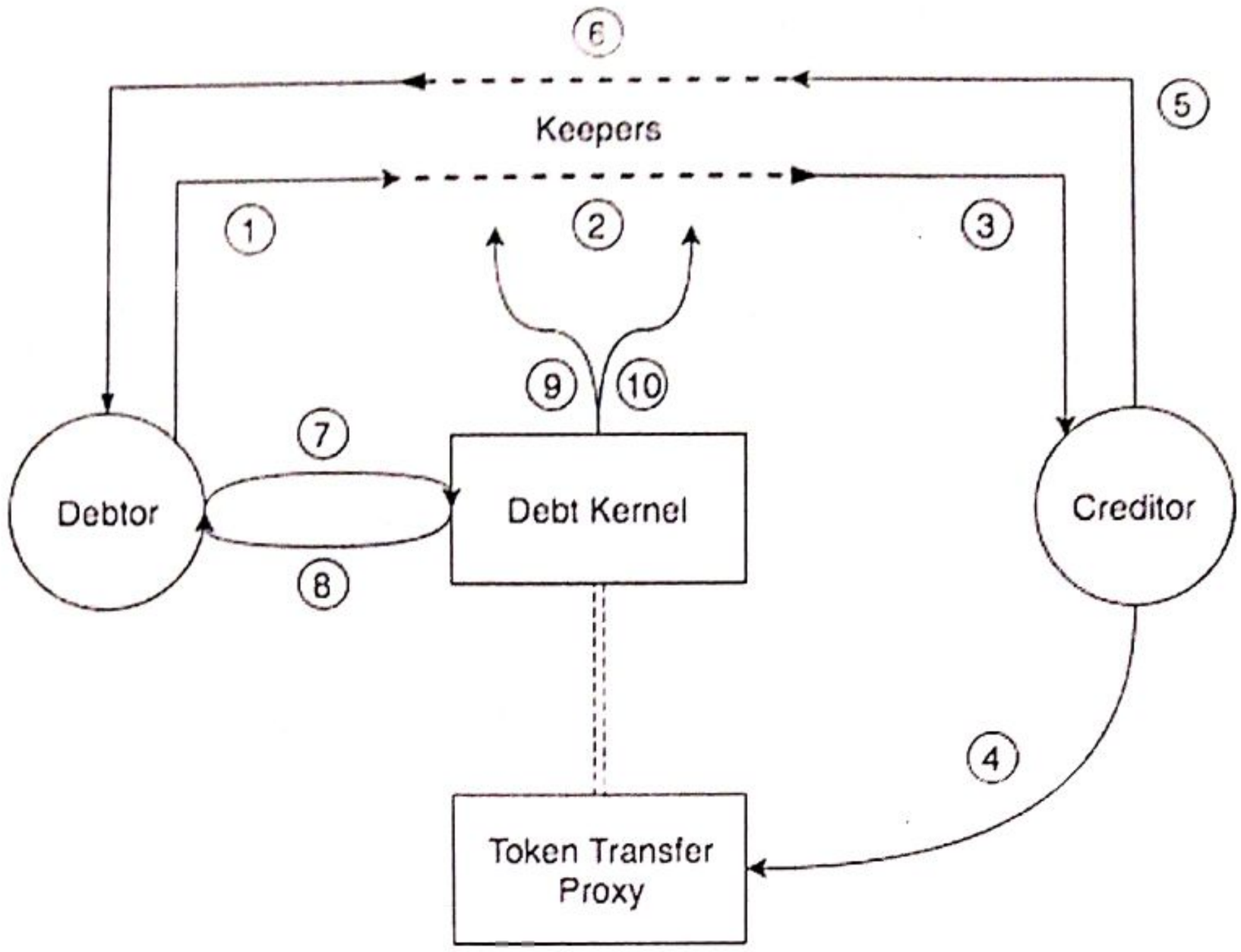

Figure 2.4: The Process of Debtor-Filler Order Submission[17]

The following steps correspond to the circled numbers in the above diagram:

1. Debtor requests loan from an underwriter.

2. Debt Order Handshake (described in detail further) occurs between the debtor, underwriter, and relayer(s), resulting in the relayer listing a valid, complete debt order.

3. Creditor evaluates the terms of the Debt Order on a relayer's public order book.

4. If the creditor wants to fill the order, he first grants the token transfer proxy an approval for transferring an amount of tokens greater than or equal to principal + creditorFee (i.e. using the ERC20 approve method).

5. The creditor then submits it directly to the Debt Kernel contract.

6. The Debt Kernel transfers an amount of principal - debtorFee from the creditor to the debtor.

7. The Debt Kernel transfers the underwriter her allotment of the fee, as defined by the Debt Order.

8. The Debt Kernel transfers the relayer his allotment of the fee, as defined by the Debt Order.

Creditor-Filler Order Submissions: The arrangements in which a creditor fills a complete order

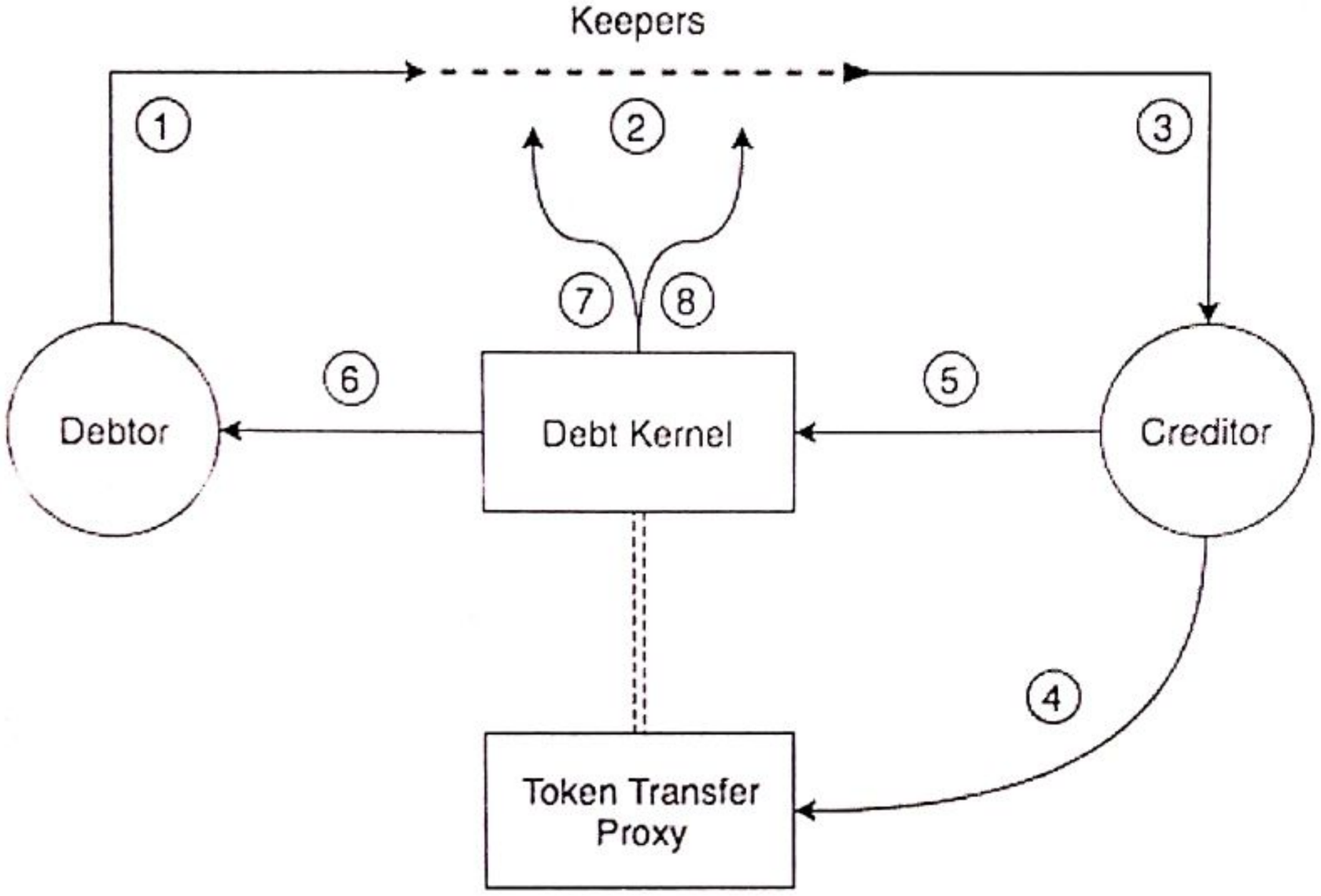

Figure 2.5: The Process of Creditor-Filler Order Submission[17]

The following steps correspond to the circled numbers in the above diagram:

1. Debtor requests loan from an underwriter.

2. Debt Order Handshake (described in detail further) occurs between the debtor, underwriter, and relayer(s), resulting in the relayer listing a valid, complete debt order.

3. Creditor evaluates the terms of the Debt Order on a relayer's public order book.

4. If the creditor wants to fill the order, he first grants the token transfer proxy an approval for transferring an amount of tokens greater than or equal to principal + creditorFee (i.e. using the ERC20 approve method).

5. The creditor then submits it directly to the Debt Kernel contract.

6. The Debt Kernel transfers an amount of principal - debtorFee from the creditor to the debtor.

7 The Debt Kernel transfers the underwriter her allotment of the fee, as defined by the Debt Order.

8. The Debt Kernel transfers the relayer his allotment of the fee, as defined by the Debt Order.

2.2 Open Research Issues

In their research paper named "The ICO phenomenon and its relationships with ethereum smart contract environment" which was added to IEEE Xplore on 29 March 2018, Gianni Fenu, Lodovica Marchesi, Michele Marchesi and Roberto Tonelli discussed the nature of ERC-20 tokens, where they mentioned that every token has a predetermined value fixed on Ether. However, fixing the value of these tokens on Ether may bring financial loss to the owner/holder of these tokens as exchange rate of Ether in dollar fluctuates.

Dharma, a decentralized protocol for peer-to-peer lending system provides uncollateralized loans. This system assesses credit risk, negotiate the loan terms and then service the loan. However, the Dharma protocol never mentions how the underwriters can assess the credit risk. Moreover, uncollateralized loan is risky for lenders as credit risk assessment may not be correct every time. Incorrect credit risk assessment will cause financial loss to lenders and people will be discouraged to be involved in the lending system.

Another lending platform known as Ethlend provides collateralized loans where the borrower must provide some amount of ERC 20 tokens to the lenders to become eligible for taking loans. The decrement is that, the loan system is based on Ether and not on paper money. Thus, if the exchange rate of Ethers fall down by the time of returning borrowed money, the lender face financial losses. Moreover, the ERC 20 token's value is predetermined and also fixed on Ether and not on paper money which may also bring financial loss to the lenders as the exchange rate of Ethers in dollars fluctuates and can fall down any time.

Another Ethereum based lending system is proposed by SHASHVAT SRIVASTAVA which is similiar to the Dharma Protocol. It also provides uncollateralized loans, but differs in the

way that it has got a robust credit risk assessment calculator and mentions how the credit risk is calculated. The calculator is able to do this by observing the entire credit history of the borrower is stored on the blockchain.But this credit risk assessment will not always turn out to be correct and there is a huge risk in case of incorrect calculation which will bring financial loss to lenders and ultimately no one will be interested to join the system.

2.3 Chapter Summary

Many systems have been proposed and implemented but none of the system fully guarantees the avoidance of financial loss of the lenders. In uncollateralized loan systems, the risk assessment may turn out to be incorrect and people will lose interest from the system. On the other hand, ERC-20 token based collateralized loan systems does not always ensure that the lenders will not face money loss.

Chapter 3

System Overview

3.1 Proposed System

To remove the first deficiency of ERC-20 Token, that is, its value is fixed on Ether and not on paper money, the code of ERC-20 token needs to be modified such that the price is fixed on USD.

To increment the value periodically, we must raise the rate of each token time to time up to a certain percentage. After reaching the threshold value, the rate will no longer be incremented.

To build a lending system which is based on paper money and not on ether, we must acknowledge the Ether equivalent of a USD. After attaining the information, we have to multiply the value by the amount of USD to be borrowed.

The system will eventually follow the steps mentioned below:

1. The system will execute an algorithm which automatically updates the price of token in Ether based on a fixed USD when a person buys some tokens.

2. Prerequisite for borrowing any amount of money is to buy ERC-20 tokens from administrator or other sellers. If a person buys some token, he/she has to buy according to the updated token price of the time of buying.

3. The system will periodically update the price of tokens a person owns up to a certain threshold value. After reaching the threshold value, the increment will stop.

4. If someone buys some tokens from a token holder, he/she must pay the extra incremented price along with the base price.

5. After buying required amount of tokens to borrow money, the borrower will put the amount of dollars he/she wants and after how many days he/she will return the money

6. The system will then update the Ether equivalent of 1 USD and multiply this rate by the amount of dollar the borrower wants.

7. If a lender wants to lend money, the required amount of tokens needed to borrow money will be transferred from borrower's account to lender's account but the tokens will be locked until the borrower fails to return the borrowed amount

8. After certain days, the system will try to fetch the amount of Ethers from the borrower's account that he/she borrowed. System will execute an algorithm which will further update the Ether equivalent of 1 USD, multiply this rate by the amount of dollar borrowed and checks to see if any extra Ether needs to be paid and add the extra amount to the borrowed amount. If there is not enough balance in the wallet, all the locked tokens in lender's wallet will be unlocked.

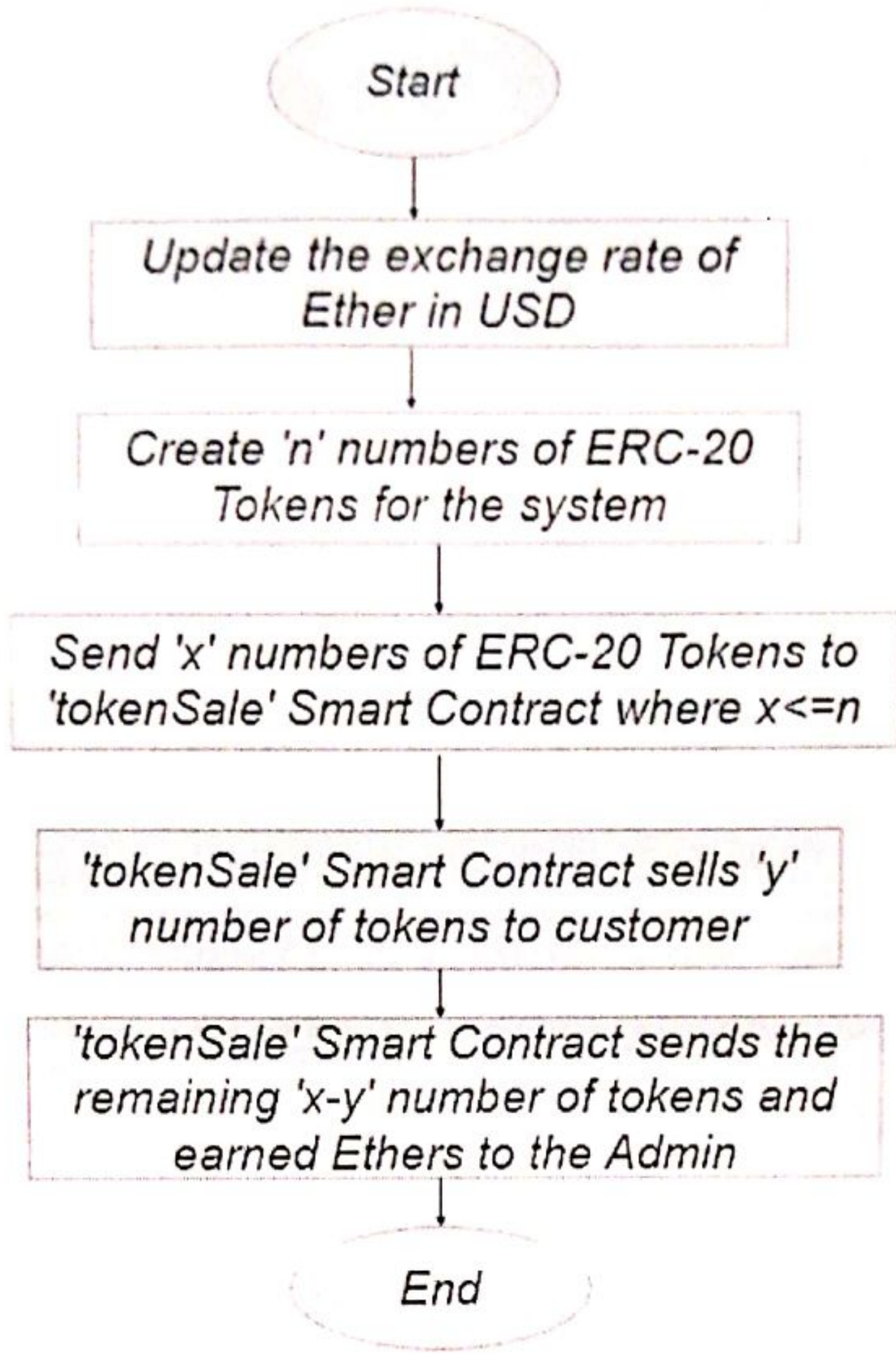

Figure 3.1: Selling Tokens to the Customer

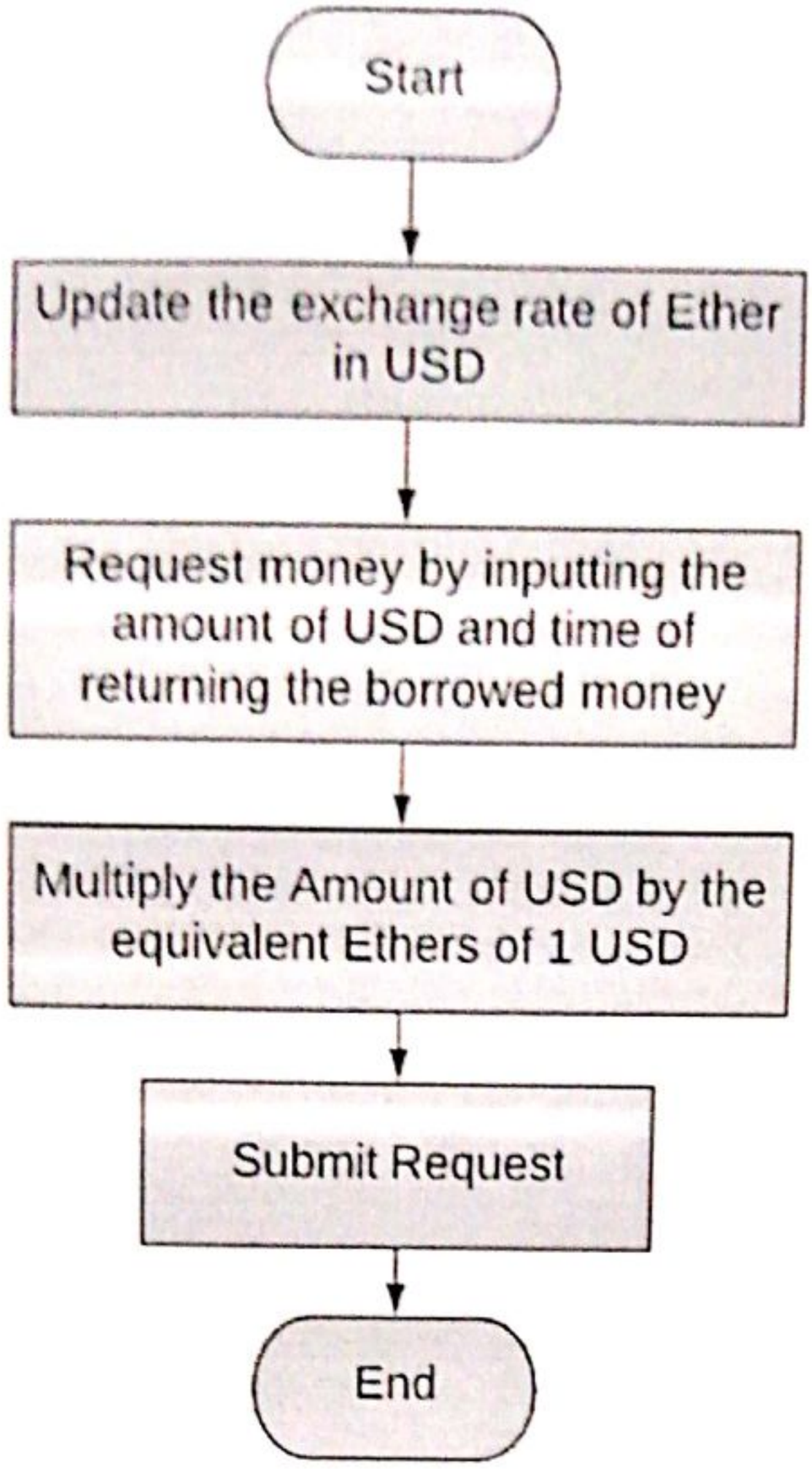

Figure 3.2: Request from the Borrower to Borrow Money

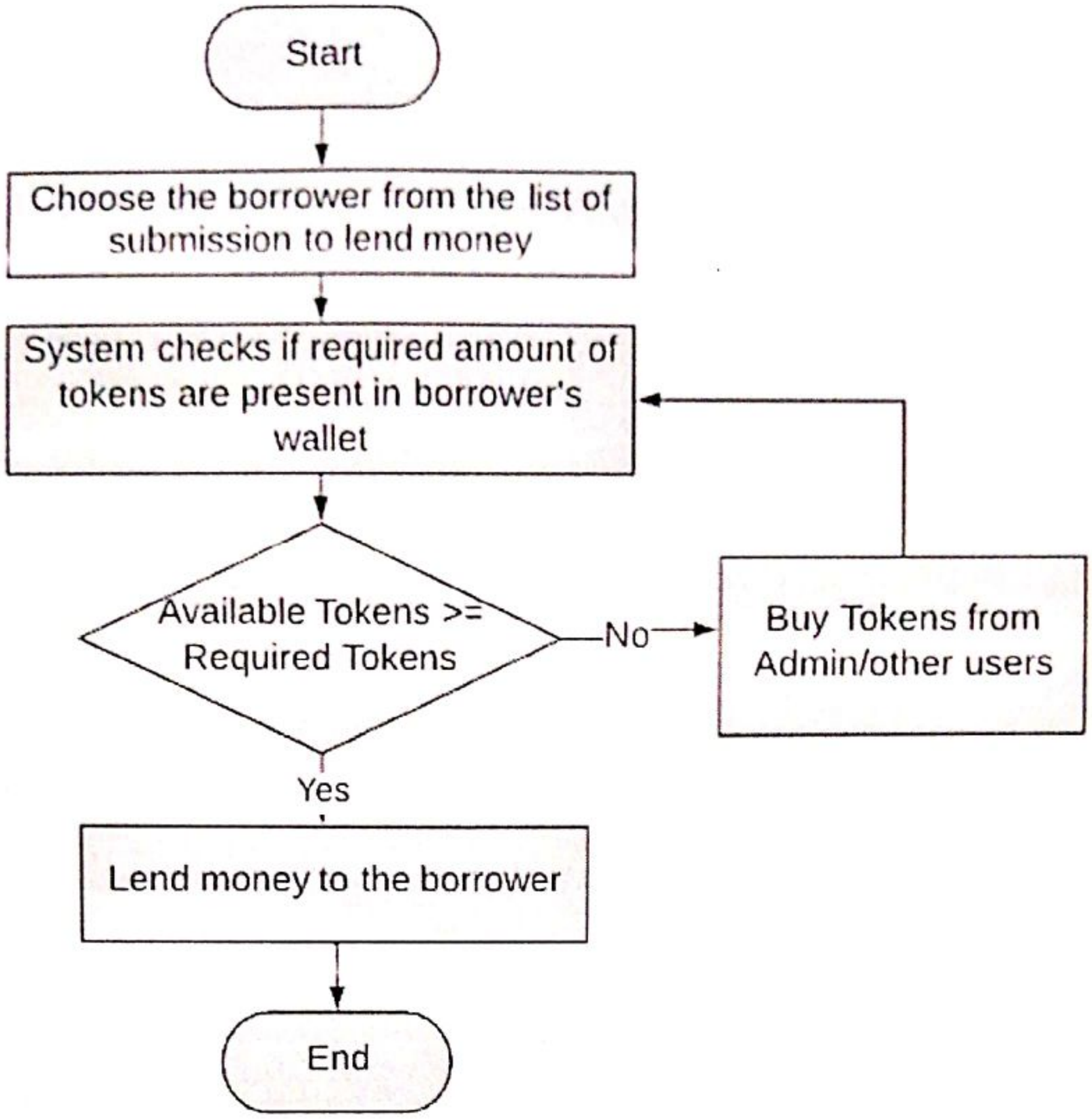

Figure 3.3: Lending Money to the Borrower

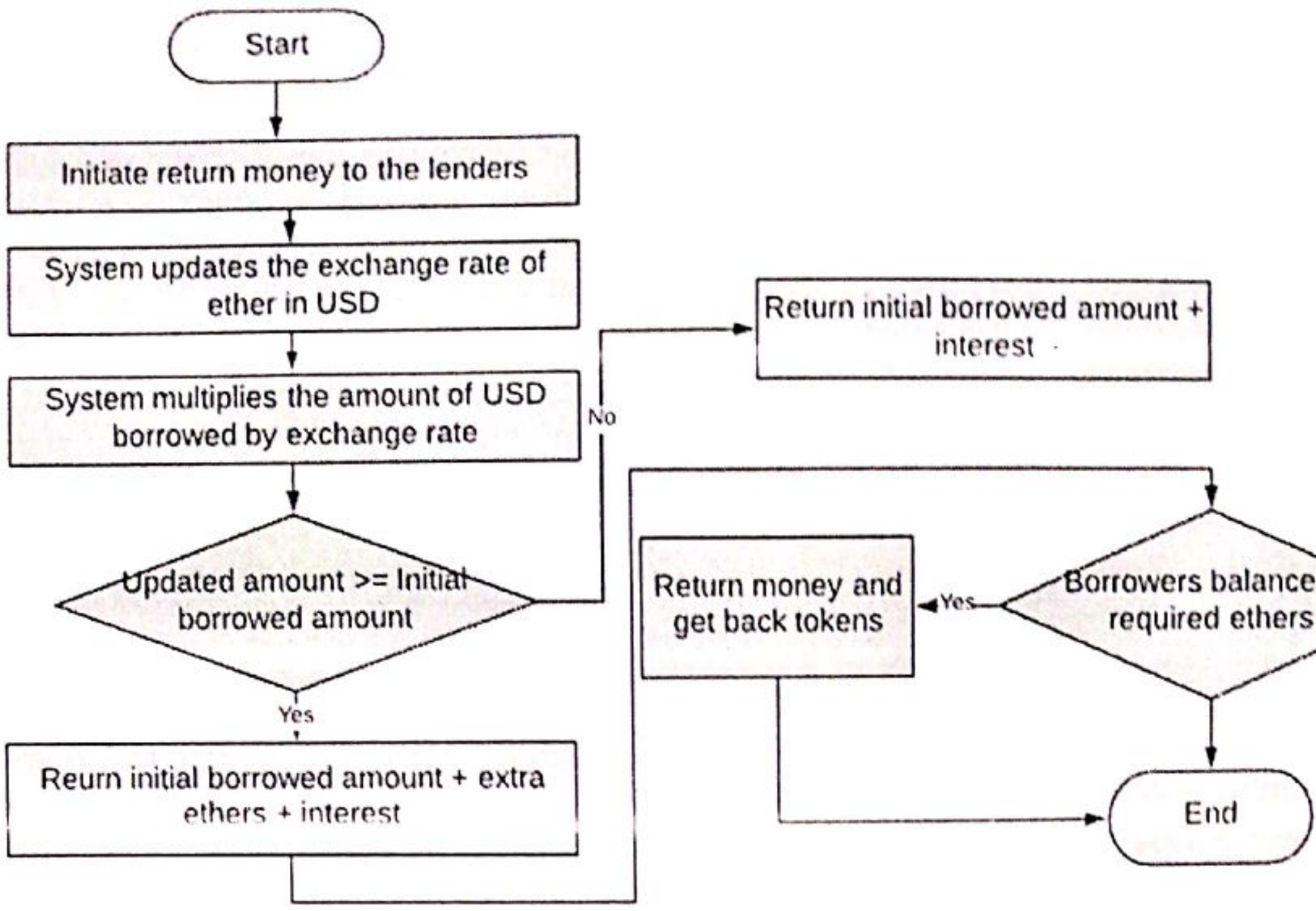

Figure 3.4: Returning Borrowed Money to the Lender

3.2 Incrementing the Value Periodically

1. After the tokens come into existence, the initial price of these tokens is stored.

2. Tokens are also assigned timestamp at the time of creation

3. According to the assigned timestamp, the tokens value is incremented according to the following formula:

   Each of the tokens value will be incremented at the rate of maximum r% for up to n years
   So, per year the tokens' value will be incremented by

   (r% / n years)

Monthly increment will be

(r% / n years) / 12 months

The extra rate which the buyer of the tokens has to pay is:

(r% / n years) / 12 months * (Initial Timestamp – Current Timestamp)

For example, 2 months after the existence of the token, the buyer has to pay an ex-

tra rate of (r% / n years) / 12 months * (2 months) along with the current updated price of the token.

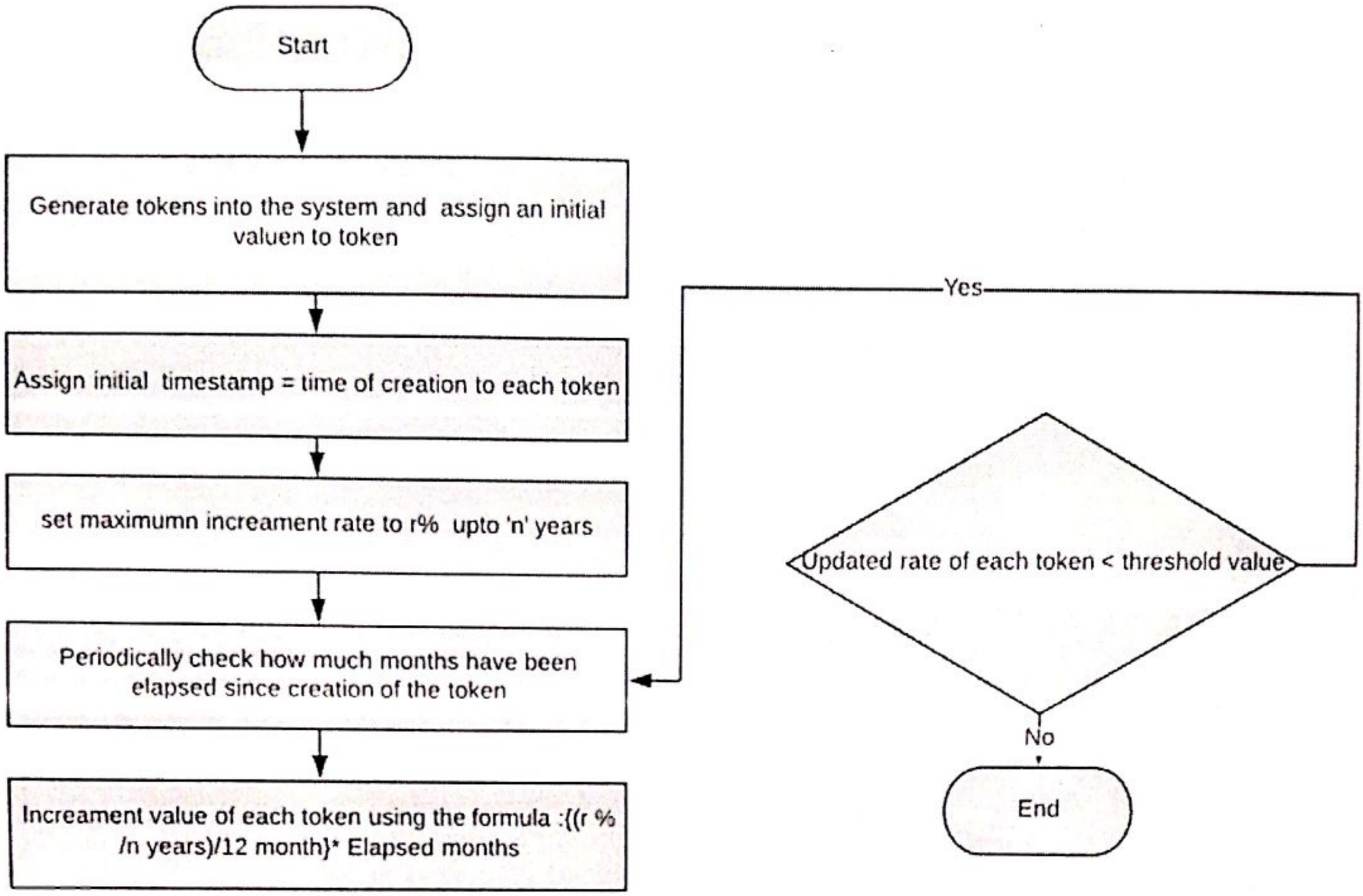

Figure 3.5: Increment of Token Value Periodically

3.3 Fixing the Token value on Paper Money

1. A scrapper made by JavaScript fetches how much dollar is equivalent to 1 Ether

2. Let's assume that 1 Ether is equivalent to d dollars.

3. Then, 1 USD is equivalent to:

   (1/d) Ethers

4. Let's assume that the price of 1 token is set to x dollars.

5. Then, the price of the token in Ether will be:

   x * (1/d) Ethers

6. As the value of d will fluctuate, x * (1/d) will also fluctuate time to time. According to the fluctuations, extra or less ethers will be required to buy a token. Smart Contract which is used for selling and buying tokens needs to be modified in the following ways:

7. The token price is not fixed at the time of contract creation.

8. Instead, the token price is set every time a buying-selling of tokens occurs

9. Instead of the constructor of the contract, buyTokens function will be responsible for setting up the token price

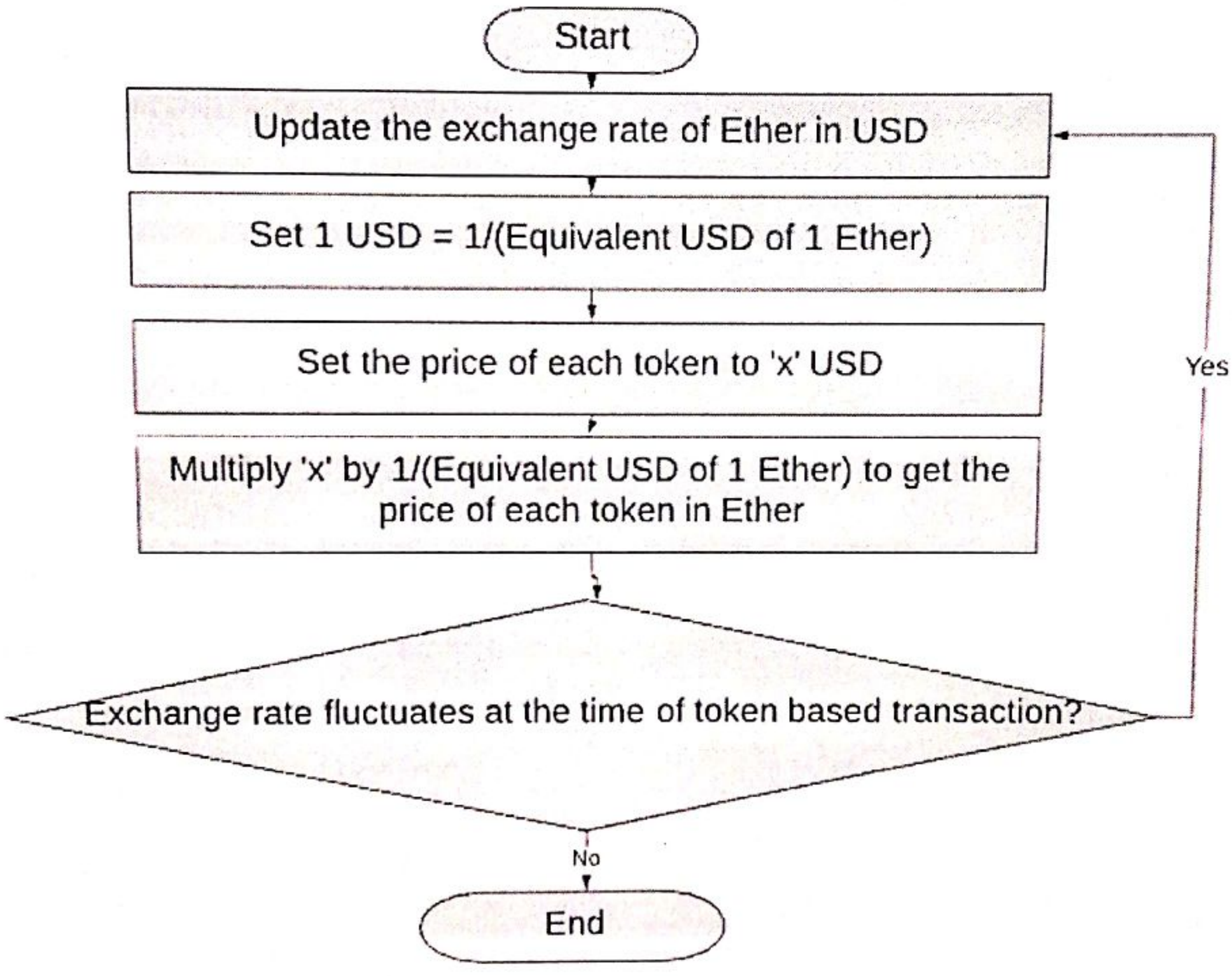

Figure 3.6. Setting the Price of ERC-20 Token on USD

3.4 Making Paper money as the lending material

1. The javascript scrapper scraps the amount of ethers which is equivalent to 1 dollar.

2. If 1 ether is equivalent to d dollars, then, 1USD equal:

(1/d) Ethers

3. If the amount of dollars which the borrower wants to borrow is b, then the amount of ethers requested will be:

$$b * (1/d) \text{ Ethers}$$

4. Borrower's also need to specify how much days he/she will take to return the money.

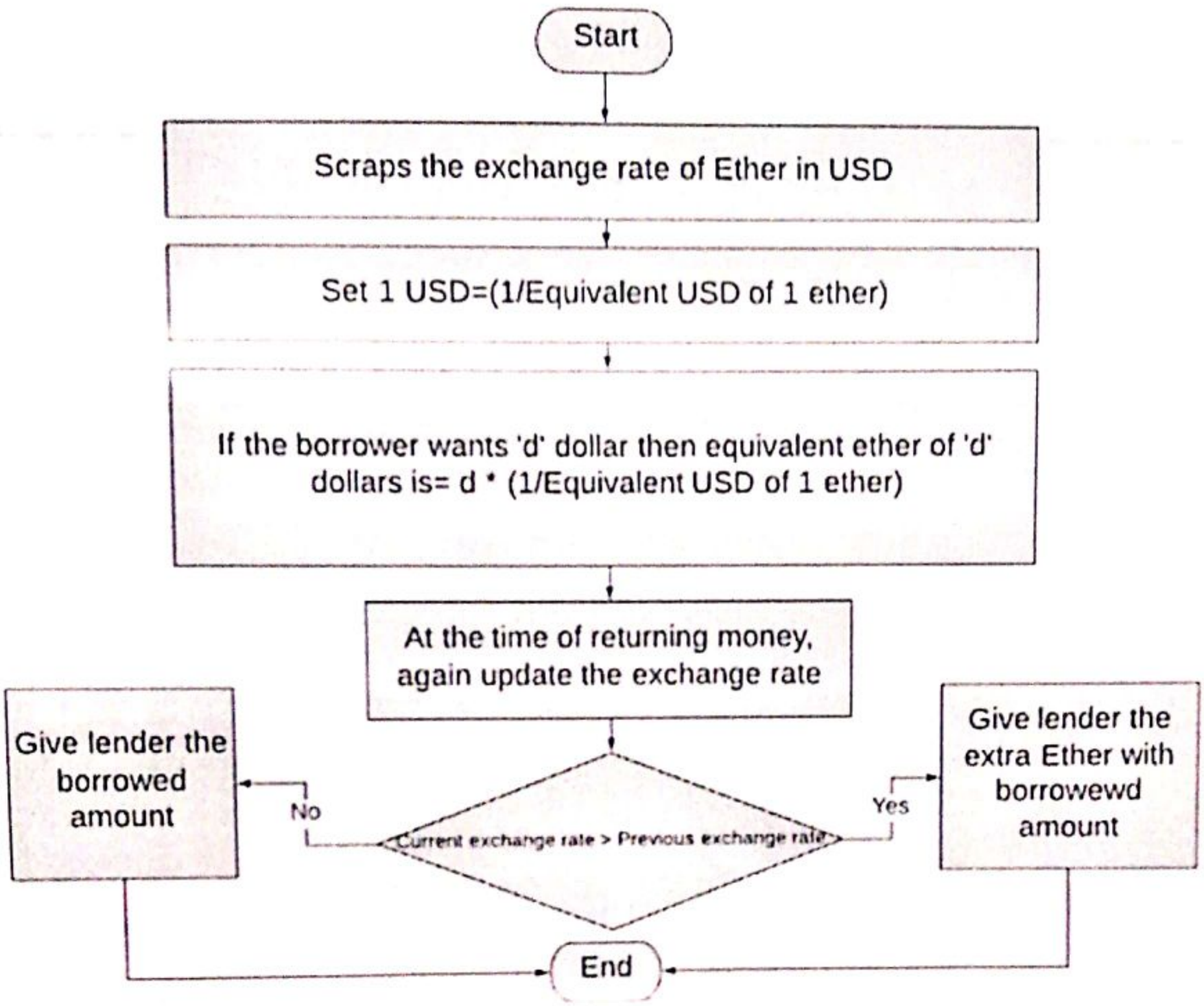

Figure 3.7: Conversion of USD to Ether for Ensuring

3.5 Fixing the Required Amount of ERC-20 Tokens For Taking Loans

1. The amount of Tokens the borrower must provide to the lenders is equal to the following equation:

   (amount of ethers borrowed/token price in ether) + 5

   From the above formula, it can be seen that the borrower must provide the token whose price is equivalent to the amount he borrowed, plus, 5 extra tokens must be provided.

2. If the borrower pledges to return the money after x days, then the token will get locked for x days inside the lender's account and lender can no longer use these tokens until borrower fails to return money and the tokens will get unlocked.

3. If the borrower successfully returns the money, then the tokens will be returned to the borrowers.

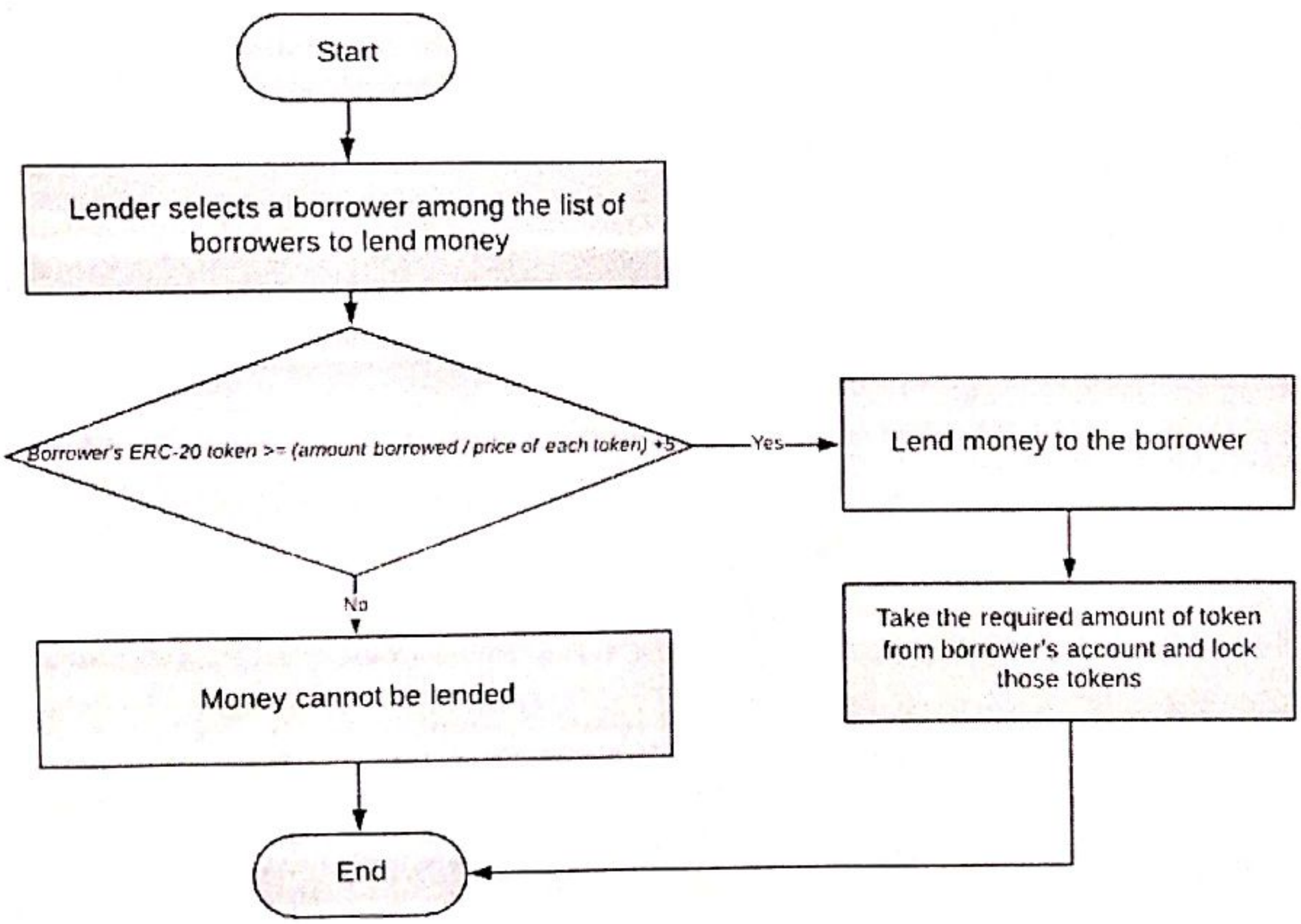

Figure 3.8: Taking ERC-20 Tokens as Collateral

Chapter 5

Result & Discussion

5.1 Overview

The proposed protocol ensures that the lender will not face any financial loss by implementing the following improvements:

1. The lending system is based on dollar and if the exchange rate of Ether in dollar increases, the borrower has to return the extra amount of Ether to compensate.

2. In addition to that, the borrower has to give a fixed amount of interest and thus the lender will be benefited.

3. ERC-20 tokens as digital assets which the borrower has to provide as pledge at the time of borrowing, ensures that the borrower will not fly away with the money he/she borrowed and causes loss to the lenders.

4. The borrower will not only provide tokens equivalent to the money he borrowed, he will have to provide 5 extra tokens to make the system more secured.

5. The system also increases the value of tokens periodically so that more people will be encouraged to enter the system as keeping the tokens in wallet will benefit that person even if he/she does not get involved in lending.

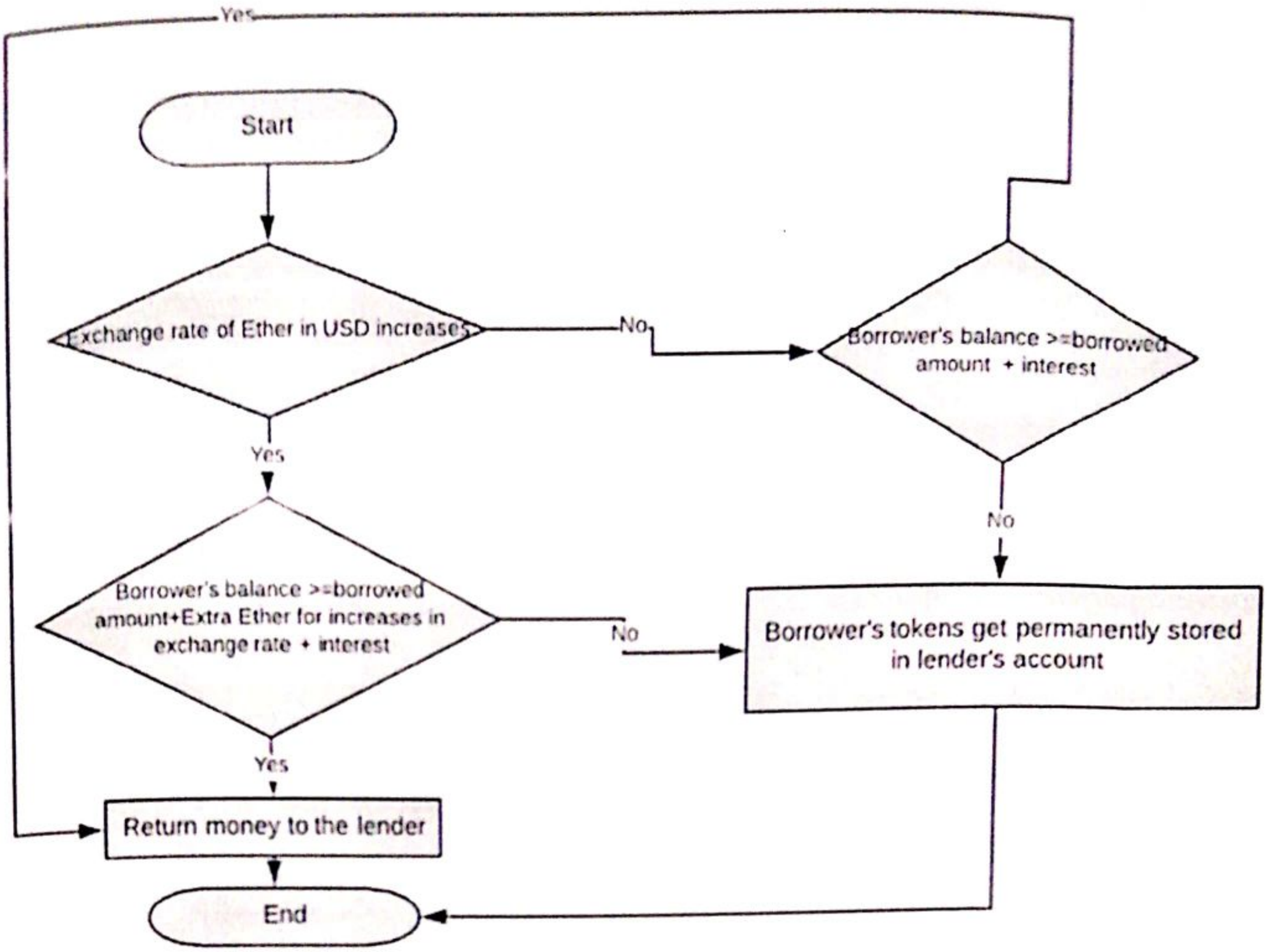

Figure 5.1: System Ensuring No Financial Loss

5.2 Performance Evaluation

5.2.1 Putting USD as Transactional Unit

1. The exchange rate of Ethers in USD is scrapped from the following link:

   https://www.coingecko.com/en/price_ charts/ethereum/usd

2. The equivalent ethers of 1 USD is then acquired by the following equation:

   (1 / Equivalent Ethers of 1 USD)

3. If a user wants to borrow 5 dollar, then the equivalent ethers to be borrowed is:

   5 USD * (1 / Equivalent Ethers of 1 USD)

4. If , at a time, 1 USD = 0.007331378299120234 Ethers, then the amount of ethers to be borrowed is:

(5 * 0.007331378299120234) Ethers

5.2.2 Incrementing the Token Price Periodically

1. If the token price will increase at the rate of 10% for 5 years, then the increment in each month will be:

(Initial Token Price at the Time of Generation) * (10% / 60 Months) * Months elapsed since token generation)

2 If the initial token price at the time of generation was 0.000036656892495601 Ether and 27 months have been elapsed, then the extra price to be paid for each token will be:
0.000036656892495601 Ether * (10% / 60 Months) * 27 Months = 0.000001583577712609 Ether

Suppose a seller wants to buy 1500 tokens from the admin whose token came into existence 27 months before the buyer is buying. The result is as following:

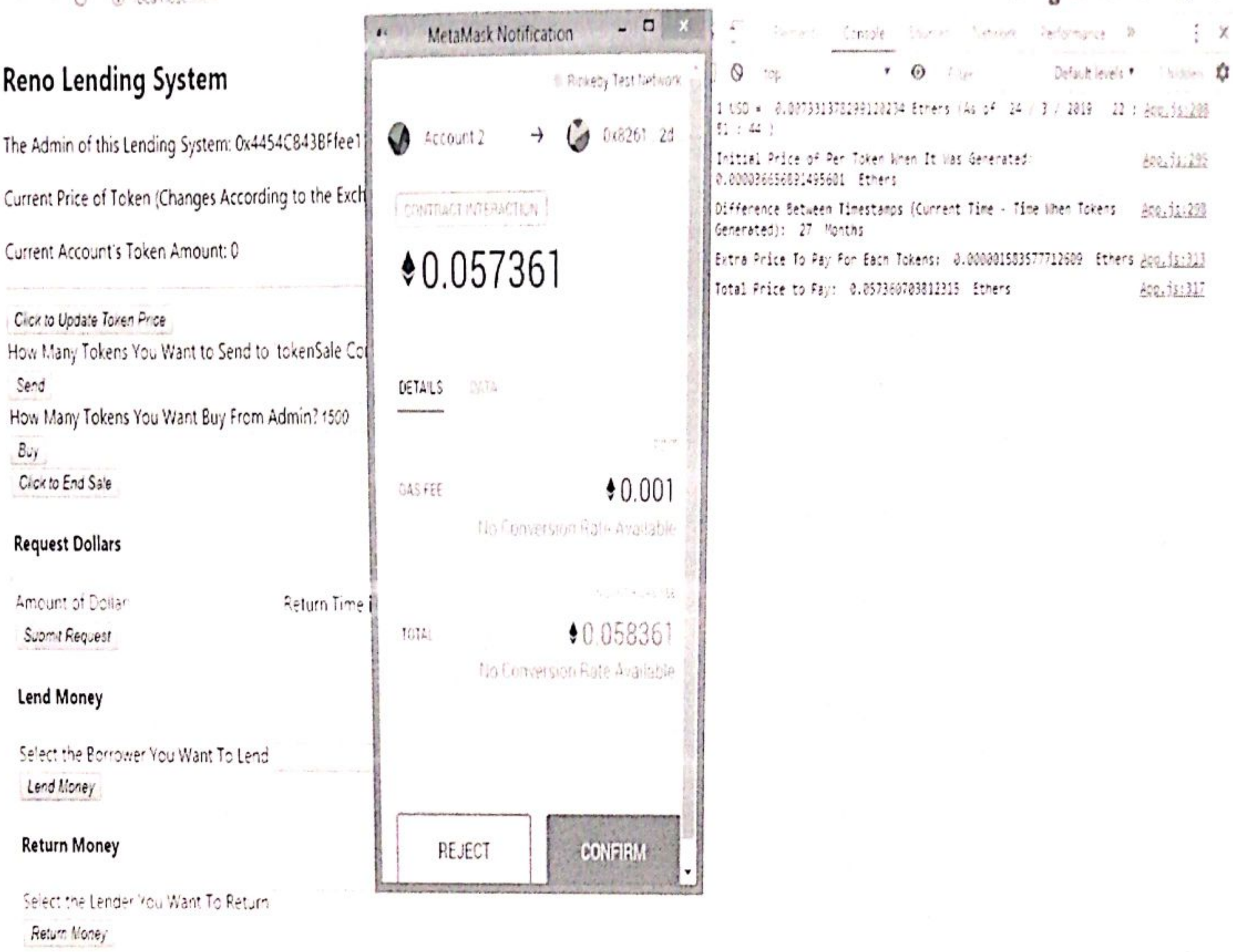

Figure 5.2: GUI Representing Periodic Increment of ERC-20 Tokens

5.2.3 Fixing the Token Price on Paper Money

1. Price of each token in USD is fixed to 0.005 USD
2. A JavaScript Scrapper is used to fetch the exchange rate of 1 Ether in USD from the following link:

   https://www.coingecko.com/en/price_ charts/ethereum/usd

3. The equivalent ethers of 1 USD is achieved by the following equation:

   (1 / Equivalent Ethers of 1 USD)

4. Price of each token will then be:

0.005 USD *(1 / Equivalent Ethers of 1 USD)

5. The system will update this value before each transaction as the exchange rate fluctuates time by time.

6. The exchange rate of 24th March, 09:34:29 pm is as follows:

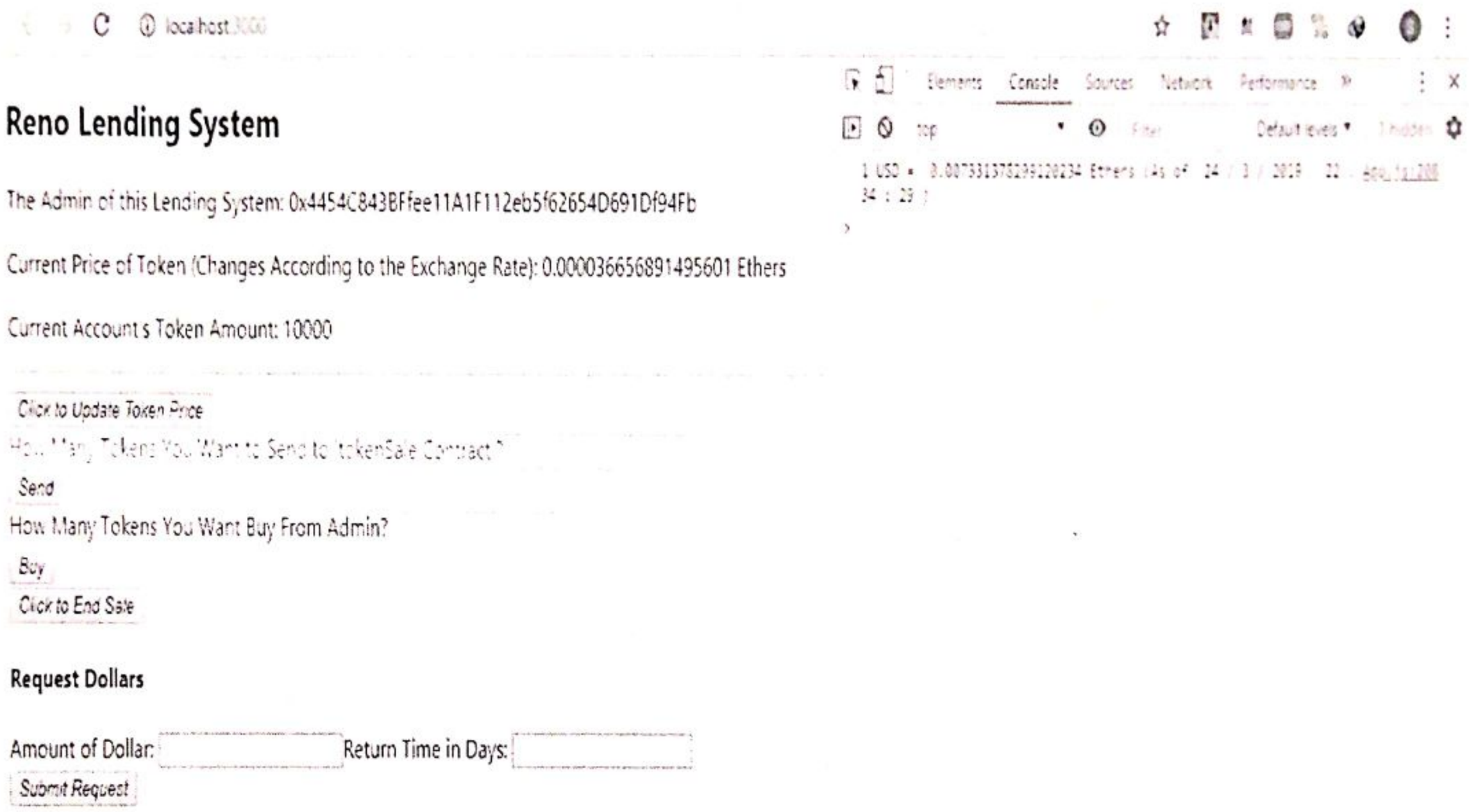

Figure 5.3: GUI Showing Updated Exchange Rate of Ether in USD

5.2.4 ERC-20 Tokens as Collateral

1. The borrower has to provide ERC-20 token worth the price of ethers to be borrowed, plus 5 tokens extra.

2. If the price of each token is 0.000036656892495601 Ether, then the number of tokens to be provided is as follows:

(Amount of Ethers Borrowed / Price of Each Token in Ether) + 5 Tokens

→((5 * 0.07331378299120234) Ethers / 0.00003665689249560l Ether) + 5 Tokens

3. The demonstration is as follows:

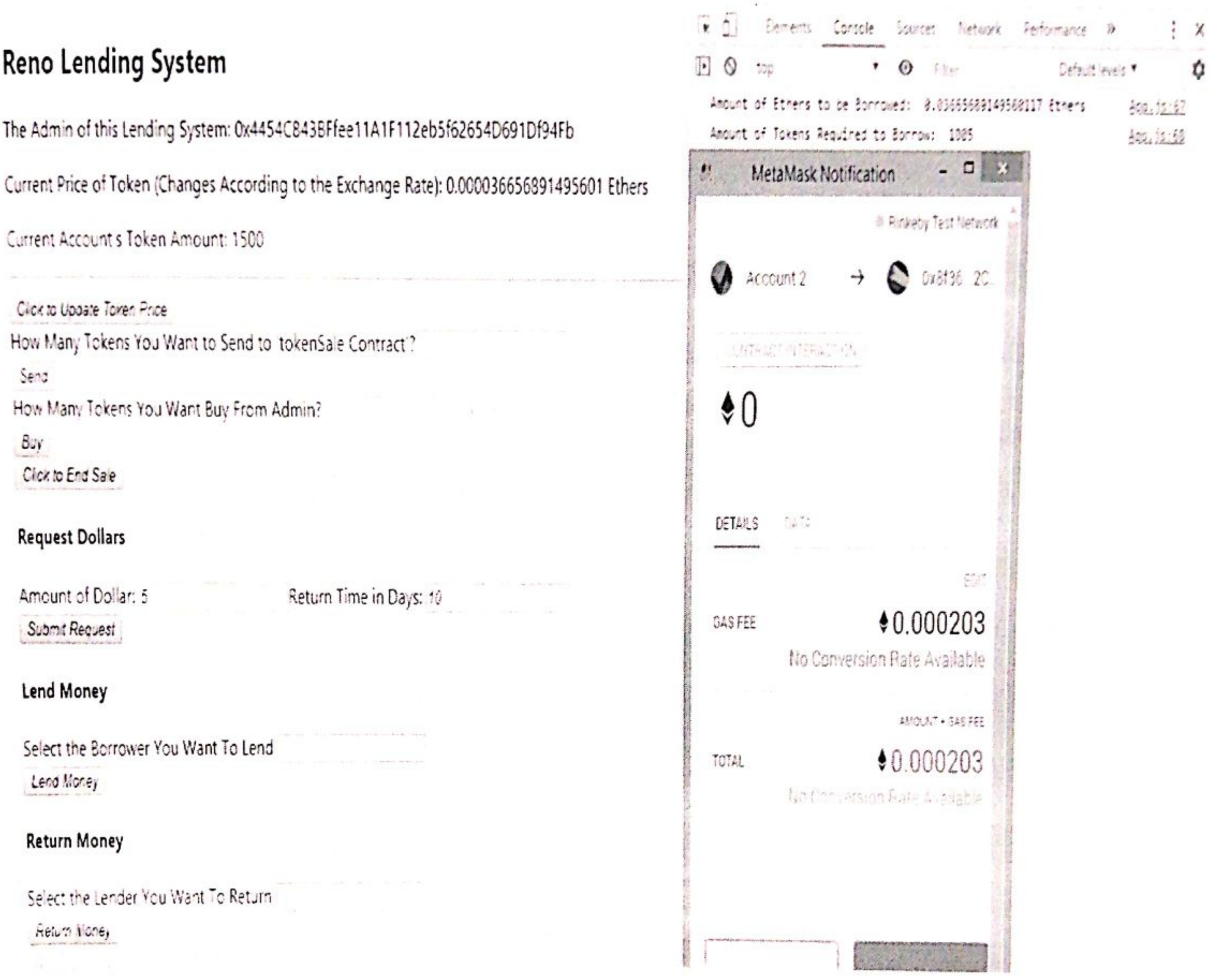

Figure 5.4: GUI Showing Required Amount of Tokens to Borrow Money

5.2.5 Getting Interest at the Time of Return and Getting Extra Ethers if Exchange Rate Decreases

1. If the interest is based on daily basis and the daily interest is 5

(Amount Borrowed * 5% * Number of days since borrowing)

2. If borrower borrows 0.036656891495601l7 Ethers for 10 days. then total interest to be

paid is:

(0.0366568914956011 7 Ethers * 5% * 10 Days)

3. In addition to that, if exchange rate fluctuates, then system manages to get extra money for compensation by the following equation:

(New Exchange Rate * Number of Dollars Borrowed) – (Amount of Ethers Initially Borrowed)

4. Total amount to be returned is as follows:

Amount Borrowed + Total Interest + Extra Ethers for Exchange Rate Fluctuations

5. The demonstration is as follows:

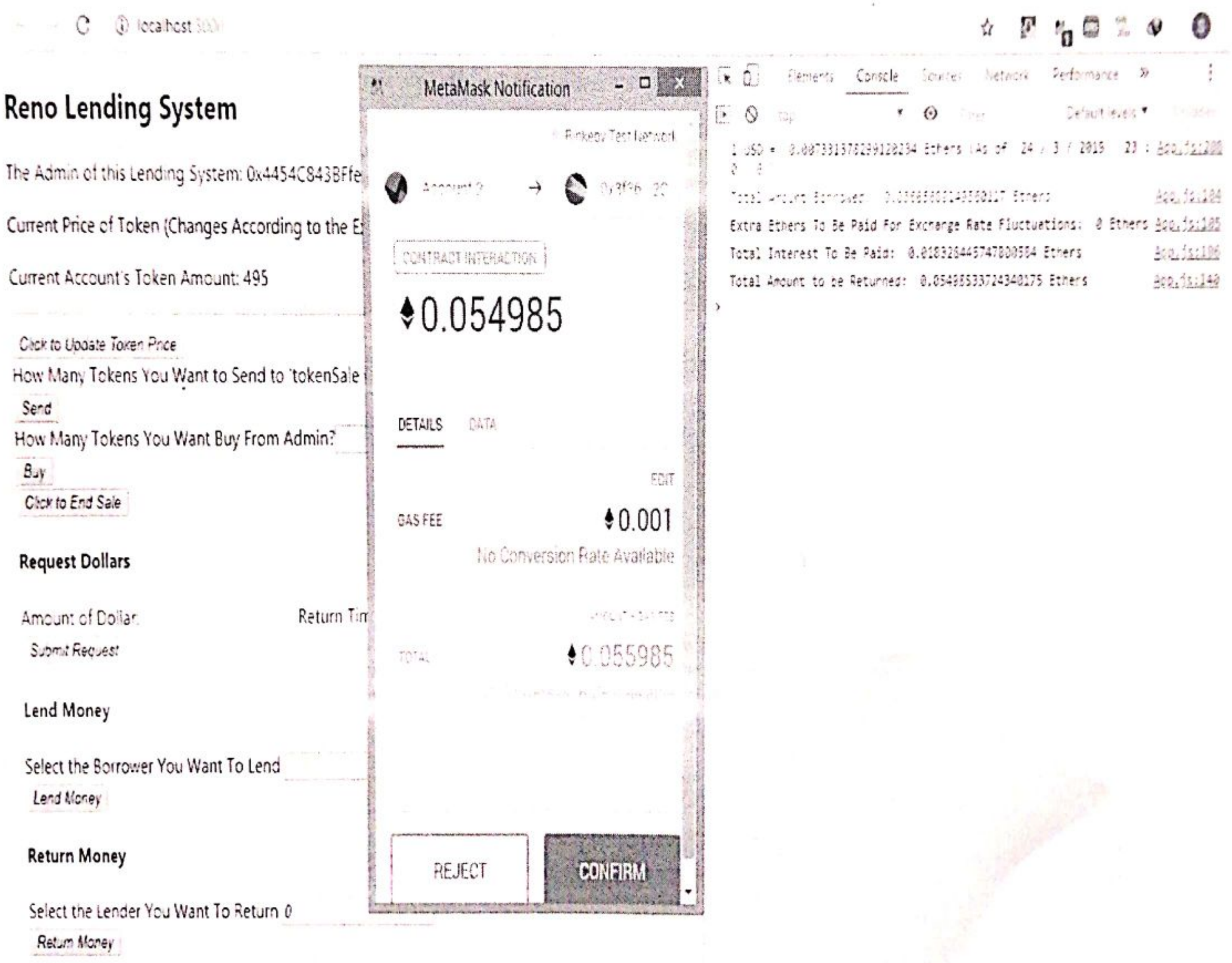

Figure 5.5: GUI Demonstrating the Amount of Ethers to be Returned

5.3 Chapter Summary

From the evaluation of the proposed system, it can be clearly seen that the system fully ensures no financial loss for the lenders. The system fixes the transactional unit on paper-money so that the fluctuations in exchange rate of ether in USD does not cause any financial loss for the lenders. The system also increments the value of ERC-20 token periodically so that the holder of these tokens get benefited and more people are encouraged to join the system. Besides the transactional unit, the system also fixes the price of tokens in USD so that the decrement in exchange rate of ether in USD does not create variations in token price with respect totime. The system takes out the required amount of tokens from the borrower's account as collateral which is almost equivalent to the money borrowed from lender, thus assuring maximum security of lender's finance.

Chapter 6

Conclusion & Future Recommendation

6.1 Conclusion

The implemented protocol, which makes sure that the lender will never face financial loss at any cost, is a clear improvement over the existing protocols. The effectiveness of the protocol is analyzed and tested and the result shows how the system is beneficial to the lenders. ERC-20 Token Standard is modified to tackle the exchange rate fluctuations of Ether and to increment each token's value periodically so that more people will be encouraged to join the system. Using USD as transactional unit ensures that the descending in the exhange rate of Ethers in USD provides extra Ethers to the lenders for the compensation.

6.2 Future Recommendation

The efficiency of the proposed system can be increased in the near future if following improvements are added:

1. Besides colleteralized loans, the facility of getting uncollateralized loans can be added. For this, an efficient risk assessment calculator has to be created which will analyse all the borrower's transaction history and based on that, an accurate estimation will be generated by the calculator

2. The system should be able to accept several types of tokens as collateral besides ERC-20 tokens. Example of popular Ethereum tokens are: ERC-223, ERC-777, Binance Coin, Maker, OmiseGo, Basic Attention Token etc.

Bibliography

[1]Santoshi Nakamoto . Bitcoin: A Peer-to-Peer Electronic Cash System . Santoshi Nakamoto Institute . Oct 31, 2008. (https://nakamotoinstitute.org/bitcoin/)

[2]Salt Technology Ltd . Salt : A Blockchain-Backed Loans White Paper IO .July 07, 2017. (https://whitepaper.io/document/85/salt-whitepaper)

[3]Elastos Foundation . Elastos : A Smart Web Powered By Blockchain White Paper IO . January 01, 2018 (https://whitepaper.io/document/45/elastos-whitepaper)

[4]Credissimo . Nexo : The World's First Instant Cypto-backed Loans. White Paper Database. May 31, 2018 (https://whitepaperdatabase.com/nexo-whitepaper/)

[5]Vitalik Buterin. Ethereum : A Next Generation Smart Contract and Decentralized Platform. Github. November 2013. (https://github.com/ethereum/wiki/wiki/White-Paper)

[6]ETHLend Organization. ETHLend.io White Paper - Democratizing Lending. Github, 25 February 2018 (https://github.com/ETHLend/Documentation/blob/master/ETHLendWhitePaper.md)

[7]Tanzo IO . Tanzo : A Blockchain-Based Social Marketplace Handmade Goods Jun 1, 2018. (https://tanzo.io/assets/pdfs/Tanzo_ Whitepaper.pdf)

[8] Sachchidanand Singh : Nirmala Singh. Blockchain: Future of financial and cyber security. IEEE Xplore, 04 May 2017 (https://ieeexplore.ieee.org/document/7918009)

[9] Dejan Vujičić, Dijana Jagodić, Siniša Ranđić. Blockchain technology, bitcoin, and Ethereum: A brief overview. IEEE Xplore, 26 April 2018 (https://ieeexplore.ieee.org/document/8345547)

[10] Gianni Fenu, Lodovica Marchesi, Michele Marchesi, Roberto Tonelli. The ICO phenomenon and its relationships with ethereum smart contract environment. IEEE Xplore, 29 March 2018 (https://ieeexplore.ieee.org/document/8327568)

[11] One Ledger IO. Oneledger: Public Blockchain Whitepaper Jun 18, 2018 (https://oneledger.io/whitepaper/oneledger-whitepaper.en.pdf)

[12] Mixin Network Organization. Mixin: A free and lightning fast peer-to-peer transactional network for digital assets. Jun 19, 2018. (https://whitepaperpagoda.com/wp-content/uploads/2018/06/Mixin-White-Paper.pdf)

[13] David Vorick , Luke Champine . Nebulous Inc . Sia: Simple Decentralized Storage November 29,2014. (https://sia.tech/sia.pdf) Nicolas Van Saberhagen . Monero : Crypto V2.0

October 17,2013. (https://whitepaperdatabase.com/monero-xmr-whitepaper/)

[15] Ethan Heilman, Leen AlShenibr, Foteini Baldimtsi, Alessandra Scafuro and Sharon Goldberg,Boston University,George Mason North Carolina State University. Eprint Publication. TumbleBit: An Untrusted Bitcoin-CompatibleAnonymous Payment Hub Jun 5,2016. (https://eprint.iacr.org/2016/575.pdf)

[16]S. SRIVASTAVA. WPI Publication. Decentralized Lending in Ethereum Cryptocurrency. February 2018 (http://users.wpi.edu/ ssrivastava2/docs/STEMthesis.pdf)

[17]Nadav Hollander. Dharma Organization. Dharma: A Generic Protocol for Tokenized Debt Insurance (whitepaper.dharma.io)