

Blockchain Transaction System

MINOR PROJECT REPORT

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR

THE AWARD OF THE DEGREE OF

BACHELOR OF TECHNOLOGY

Information Technology



Submitted By:

Rohit Samal (1805547)

Rajat Singh Chauhan (1805544)

Tarandeep Kaur (1805567)

Submitted To:

Prof. Mohanjit K Kang

Assistant Professor

Minor Project Coordinator

Department of Information Technology

Guru Nanak Dev Engineering College,

Ludhiana-141006

Abstract

“Blockchain Transaction System” is a project implementing “Blockchain Technology” in the transaction system to make system without a central authority. This system provides users to make peer-to-peer transaction of digital currency directly with each other removing any third-party for validation, managing and confirming the transaction. In this system transactions are immutable i.e once a transaction added to blockchain it can not be modified or tampered. This makes the system secure and free from fraud of manipulating transaction. Our project has two web application one for “Blockchain Frontend” for miners to mine the blocks in blockchain for storing transaction and other is “Blockchain Client” for users to generate wallets and making transaction.

ACKNOWLEDGEMENT

We are highly grateful to Dr. Sehajpal Singh, Principal, Guru Nanak Dev Engineering College (GNDEC), Ludhiana, for providing this opportunity to carry out the minor project work. The constant guidance and encouragement received from Prof. Mohanjit kaur kang, IT Department, GNDEC Ludhiana has been of great help in carrying out the project work and is acknowledged with reverential thanks. We would like to express a deep sense of gratitude and thanks profusely to Prof. Mohanjit Kaur Kang without her wise counsel and able guidance, it would have been impossible to complete the project in this manner. We express gratitude to other faculty members of Information Technology Department of GNDEC for their intellectual support throughout the course of this work. Finally, we are indebted to all whosoever have contributed in this report work.

List of Figures

1	Authentication Processes for Transaction on the Blockchain	15
2	Blocks are chained together using the previous block's hash to form a Blockchain	15
3	Resolving Conflicts	16
4	Blockchain Client UI	16
5	Blockchain Frontend UI	17
6	Python	19
7	Blockchain Client Dashboard	21
8	Generating wallets containing public and private key	22
9	Drop down menu of dashboard	22
10	Dashboard for Transaction details form	23
11	Filled form of transactions details	24
12	Confirm Transaction Dialogue Box	25
13	Showing Successful Transaction Message	26
14	Blockchain Frontend	27
15	Adding Node in the Node URLs	28
16	Node added for retrieving	29
17	Showing Transaction to be added on the block	30
18	Transaction added on the Blockchain	31
19	Adding node to the Node URL	32
20	Transaction can be viewed by the client	33

List of Tables

Title page	
Abstract	i
Acknowledgement	ii
List of Figures	iii
List of Tables	iv
Table of Contents	v

Contents

1 Introduction	8
1.1 Introduction to Project	8
1.2 Project Category	8
1.3 Objectives	8
1.4 Problem Formulation	8
1.5 Existing System	9
1.6 Proposed System	9
1.7 Unique Features of the System	9
2 Requirement Analysis and System Specification	10
2.1 Feasibility Study	10
2.2 Software Requirement Specification	11
2.3 Validation	13
2.4 Expected hurdles	13
2.5 SDLC Model Used:	13
3 System Design	14
3.1 Design Approach	14
3.2 Detail Design	14
3.3 System design	14
3.4 User Interface design :-	16
3.5 Methodology	17
4 Implementation, Testing and Maintenance	19
4.1 Introduction to Language, IDE and Tools	19
4.2 Testing Techniques and Test plans:-	20

5	Result and Discussion	21
5.1	Snapshot of system with brief detail of each :-	21
6	Conclusion and Future Scope	34
6.1	Conclusion	34
6.2	Future Scope	34
7	References/Bibliography	35

1 Introduction

1.1 Introduction to Project

These days , there are lot of transactions of digital currency are ongoing. Even it is for a large scale business or for a single person to buy its daily needs. These transactions involves the third parties like banks for their management, validation and confirmation. These third parties charged us for these works. So our projects is to remove these third parties from the transaction system. Our project is to implement the Blockchain Technology in the modern transaction systems. Blockchain is a decentralized database that the data which stored in the blocks of blockchain is not managed by third parties and protected with the use of many secure cryptography hashing algorithm. Our system makes the transaction system immutable i.e after the transaction confirmed it is added to the blocks of the Blockchain and can never be modified or tampered. Our system has the function of mine which mines the block for storing these transactions. Miners have to mine the blocks by solving some cryptographic algorithm. Our project includes two web application one for blockchain miners and other for clients who makes the transactions. Our system can runs on a Web server as well as on localhost.

1.2 Project Category

Our project is an internet based application. It contains a two web application one for blockchain miners and others for users . Both web application runs on server using “Flask” a python web framework for creating web application.

1.3 Objectives

- To achieve decentralization in the modern transaction system.
- To provide user a peer-to-peer transaction system i.e sender can send receiver some digital currency directly.
- To remove the third parties involves in modern transaction system.
- To improve the security of a transaction system.

1.4 Problem Formulation

- The transaction of digital currency is managed by the third parties.
- They charged the user for managing their transaction.
- The chances of fraud are there as these transactions are mutable as they can be tampered.
- The peer-to-peer connection for transaction is not in the existing transaction system.

1.5 Existing System

- Bitcoin :- Bitcoin is a digital currency, without a central bank or single administrator, that can be sent from user to user on the peer-to-peer Bitcoin network without the need for intermediaries.
- Ethereum :- Ethereum is an open-source , decentralized blockchain Technology. Ethereum native coin is called ether. This coin is one of the largest cryptocurrencies by market capitalization after Bitcoin.
- Ripple
- LiteCoin

1.6 Proposed System

- This system is designed for users who want to have peer-to-peer transaction system.
- This system removes the third parties involved in a modern transaction system.
- This system provides users a high security for their transactions.

1.7 Unique Features of the System

- Uses various cryptographic algorithm for secure transaction.
- Uses blockchain decentralized database for storing transaction.
- Easy implementation on different platforms.
- System UI is user friendly. As a user can easily use this system with some directions.
- Generate wallets using private and public key.
- Uses nodes to fetch transaction from the server.

2 Requirement Analysis and System Specification

2.1 Feasibility Study

The feasibility study of any system is mainly intended to study and analyze the proposed system and to decide whether the system under consideration will be viable or not after implementation. That is it determines the usability of the project after deployment. To come to result a set of query is answered keeping the efficiency of the software and its impact on the domain for which it was developed. Its main emphasis is on the following three questions listed below as:

- What are the user's requirements and how does a candidate system meet them?
- What resources are available for the proposed systems? Is it worth solving the problem?
- What is the likely impact of the proposed system on the organization i.e. how does the proposed system fit within the organization?

Thus since the feasibility study may lead to commitment of large resources, it becomes necessary that it should be conducted competently and no fundamental errors of judgment are made. Different types of feasibility study and the way I performed on my project "Toxic Comment Classification".

TYPES OF FEASIBILITY STUDY

- **Economic Feasibility:** The developed system is economic feasible as for implementing this system most of the software used are free of cost. The benefits that the system provides are totally cost effective. As it is developed using Built-in packages which are provided by the python programming language. The web application is developed using Html, CSS & JavaScript, all are open sources and can be used on any platform which makes the system economic feasible.
- **Operational Feasibility:** . No major training and new skills are required as it is based on python. It will help in the time saving and fast processing and dispersal of user request and application. This system is operationally feasible as there is no special training for user to use this system and whatever little instructions on the system is required can be done so quite easily and quickly as it is essentially. **User Support:** User involvement in the building of present system is to keep in mind the user specific requirement and needs. User will have control over own information. Data can be easily fetched by the particular user. In comparison to the existing manual system, the new system can be considered to be operationally feasible. •
- **Technical Feasibility:** For creating the program for the system we use python language. As python language is open source it is easily available. For our system to connecting to the web we use Flask a python library web framework for creating web application. It creates a temporary server for running

our system. For designing our web application we use HTML, CSS & JavaScript. For adding response to our web application we use pre-built libraries like Bootstrap, JQuery, DataTables. These all are easily available and can be used directly in the web designing. So all of this makes our project technically feasible.

2.2 Software Requirement Specification

A software requirements specification (SRS) is a detailed description of a software system to be developed with its functional and non-functional requirements.

Data Requirements :-

The data to be used by this system is generated by the user. User provides transaction details through a form in the system. Data generated by the system is mostly in Json format. Data for the transaction are shown in the tables.

- Functional requirements :-

- Users should be able to generate wallets containing private and public key.
- Users should be able to generate and confirm transaction. They should be able to view confirmed transaction.
- Miners should be able to mine the blocks for storing the transaction and successfully add these blocks to the system.
- Miners should be able to fetch data from the nodes.

- Performance Requirements :-

- The web application should load and be usable within 3 seconds.
- The application should update the interface on interaction within 2 seconds.
- “Blockchain Frontend” should be able to fetch data from nodes.
- Confirmation of transaction should not take more time.

- Security Requirements :-

- Only the Rest API should be able to connect to the system.
- No transaction has to be mutable • Private key of the users should not be visible.
- No modification of the transaction should be possible by the miners like amount, receiver address etc.

- External Interface Requirement :-

- Hardware :- It includes :-

- * Intel core i5 computer
- * 512 Mb RAM
- * Hard Disk :1TB
- * Android Device with latest browser
- Miners :- Those who mined the blocks by solving cryptographic puzzles.
- The technical specifications of requirements for the software as follows:-
 - Python latest version
 - Flask latest version Html,
 - CSS,
 - JavaScript Bootstrap,
 - JQuery,
 - DataTables
- Dependability Requirements :-
 - Availability:- The system should be ready for correct service.
 - Reliability :- The system should continuity of correct services.
 - Safety :- In the system , there should be absence of catastrophic consequences on the user and the environment
 - Maintainability :- The system should have ability for easy maintenance.
- Look and Feel Requirements :-
 - System should be user friendly
 - Users should feel that they can rely on it and trust it.
 - Web application of System should be professional and user friendly.
 - System should be easy to used by the users.
- Software Quality Attributes
 - Correctness-The system should be correct in accordance with the above mentioned functional requirements.
 - Learnability-The User Interface should be clear and simple .The system should be properly documented

- Robustness-There should be less frequent errors, it should reduce the impact of operational mistakes, erroneous input data and hardware errors.
- Maintainability-It must be suitable for debugging(localization and correction of errors) and for modification and extension of functionality.

2.3 Validation

It is the process of checking that a software system meets specifications and that it fulfills its intended purpose. It may also be referred to as software quality control. It is normally the responsibility of software testers as part of the software development lifecycle. Software validation checks that the software product satisfies or fits the intended use (high-level checking), i.e., the software meets the user requirements, not as specification artifacts or as needs of those who will operate the software only but as the needs of all the stakeholders (such as users, operators, etc.). There are two ways to perform software validation: internal and external.

2.4 Expected hurdles

Expected hurdles was a functional requirements i.e “Confirm Transaction” of the system isn’t working on a android devices. It cannot take a localhost as node for Blockchain. One more hurdle was that localhost server cannot be accessed on the android device browsers.

2.5 SDLC Model Used:

SOFTWARE DEVELOPMENT LIFECYCLE (SDLC) is a systematic process for building software that ensures the quality and correctness of the software built. SDLC process aims to produce highquality software that meets customer expectations. The system development should be complete in the pre-defined time frame and cost. SDLC consists of a detailed plan which explains how to plan, build, and maintain specific software. Every phase of the SDLC life cycle has its own process and deliverables that feed into the next phase.

Why SDLC ?

- It offers a basis for project planning, scheduling, and estimating
- Provides a framework for a standard set of activities and deliverables
- It is a mechanism for project tracking and control
- Increases visibility of project planning to all involved stakeholders of the development process
- Increased and enhance development speed
- Improved client relations
- Helps you to decrease project risk and project management plan overhead

3 System Design

3.1 Design Approach

The project uses the function oriented design as this project focusing attention to the function of the program. It is based on the stepwise refinement. Stepwise refinement is a top-down strategy where a program is refined as a hierarchy of increasing levels of details .

We start with a high level description of what the program does. Then in each step, we take one part of our high level description and refine it. Refinement is actually a process of elaboration. The process should proceed from a highly conceptual model to lower level details. The refinement of each module is done until we reach the statement level of our programming language.

3.2 Detail Design

- Register node
- Generate wallets
- Users filling transaction details
- Confirming transaction
- Add nodes to transaction
- Miner mines the block
- System verifies the transaction
- Transaction added to block
- Block connect to the last block in blockchain
- Resolve conflicts between nodes
- Miner gets reward
- Transaction successful
- Users can view the transaction

3.3 System design

The systems design of this study is given by:

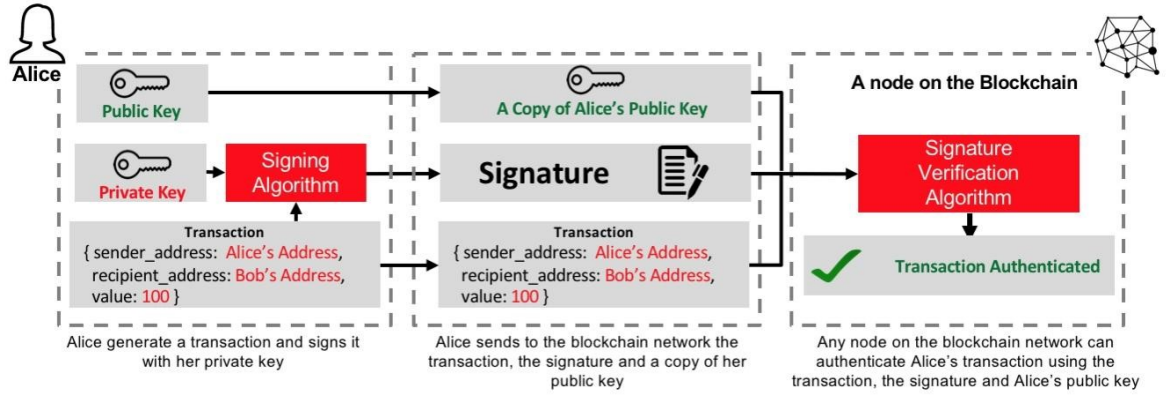


Figure 1: Authentication Processes for Transaction on the Blockchain

Transactions are grouped in blocks and blocks are appended to the blockchain. In order to create a chain of blocks, each new block uses the previous block's hash as part of its data. To create a new block, a miner selects a set of transactions, adds the previous block's hash and mines the block in a similar fashion described above. Any changes to the data in any block will affect all the hash values of the blocks that come after it and they will become invalid. This give the blockchain its immutability characteristic.

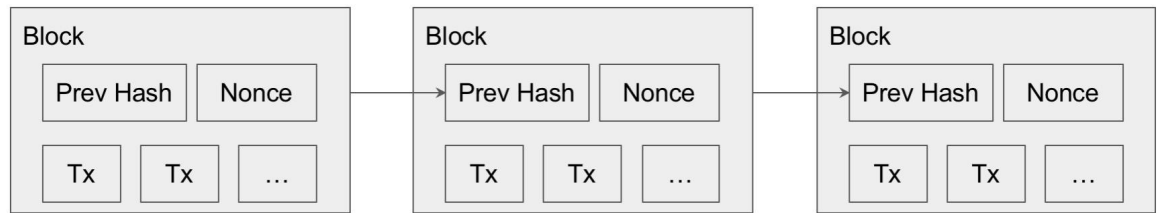


Figure 2: Blocks are chained together using the previous block's hash to form a Blockchain

If 2 miners solve a block at almost the same time, then we will have 2 different Blockchain in the network, and we need to wait for the next block to resolve the conflict. Some miners will decide to mine on top of blockchain 1 and others on top of blockchain 2. The first miner to find a new block resolves the conflict. If the new block was mined on top of blockchain 1, then blockchain 2 becomes invalid, the reward of the previous block goes to the miner from blockchain 1 and the transactions that were part of blockchain 2 and weren't added to the blockchain go back to the transactions pool and get added to the next blocks. In short, if there is a conflict on the blockchain, then the longest chain wins.

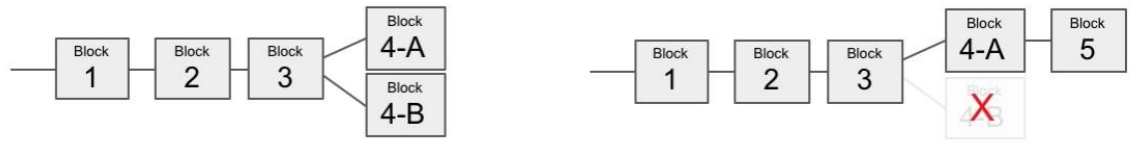


Figure 3: Resolving Conflicts

3.4 User Interface design :-

Our project UI is designed using HTML , CSS & JavaScript. For applying response to our UI we used Bootstrap a CSS framework, JQuery for JavaScript plugin and DataTables for viewing transaction details in table format.

Blockchain Client
Wallet Generator
Make Transaction
View Transactions

Wallet Generator

Click on the button below to generate your blockchain wallet

Generate Wallet

Public Key:

Private Key:

Figure 4: Blockchain Client UI

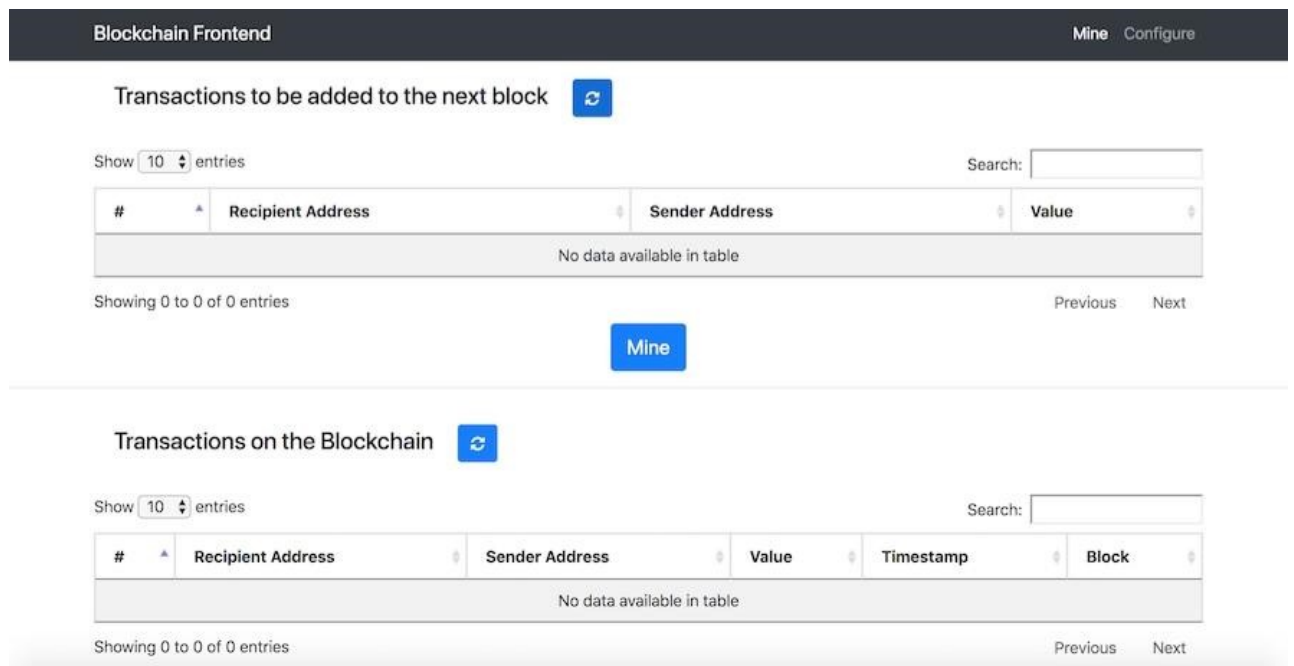


Figure 5: Blockchain Frontend UI

3.5 Methodology

Blockchain technology works by creating an environment that is secure and transparent for the financial transactions of virtual values such as Bitcoin. Hash codes of each block keep records safe in the blockchain. This is mainly because irrespective of the size of the information or document, the mathematical hash function provides a hash code of the same length for each block. So, attempting to change a block of information would generate a completely new hash value. The blockchain network is a decentralized structure that consists of scattered nodes (computers) that inspect and validate the authenticity of any new transactions that attempt to take place. This combine agreement is done through several consensus models by the process of mining. The process of mining demonstrates that each node trying to add a new transaction has gone through and solved the complex computational puzzle through extensive work and deserves to get a reward in return for their service. For the validation of a transaction, the network must confirm the following conditions: The sender account holds sufficient balance that it intends to transfer. The amount intended to transfer has not already been sent to some other recipient. Once a transaction has been validated and agreed upon by all the nodes, it then gets added to the digital ledger and protected using cryptography that uses a public key that is accessible to all the other nodes and a private key that must be kept secret. To maintain the transactions using digital currency in the blockchain network, we need to have an understanding of the digital wallet which is used to store, send, and receive digital currency. A digital wallet or a cryptocurrency wallet is a string of letters and numbers forming a public address associated with each block in the blockchain. This public address is used whenever a transaction takes place; that is, the currency is assigned to the public address of

the specific wallet. However, to prove the ownership of the public address there is a private key associated with the wallet that serves as the user's digital signature that is used to confirm the processing of any transaction. The user's public key is the shortened version of his private key generated through complex and advanced mathematical algorithms . For example, let us consider someone is trying to send you some digital currency, as the transaction is being processed, the private key in your wallet should match the crucial public address of your wallet that the currency has been assigned to. If both these keys match, then the digital currency amount is transferred to the public address of your wallet

4 Implementation, Testing and Maintenance

4.1 Introduction to Language, IDE and Tools

Python: Python is an interpreted, high-level and general-purpose programming language. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is meant to be an easily readable language. It uses English keywords where other languages use punctuation. Unlike many other languages, it does not use curly brackets to delimit blocks, and semicolons after statements are optional. Python is a popular programming language. It is used for web development, software development and system scripting.



Figure 6: Python

- Python works on different platforms.
 - Python can be used on a server to create web applications
 - Python can connect to database systems; it can also read and modify files
 - Python can be used to handle big data and perform complex mathematics
 - Python runs on an interpreter system, meaning that code can be executed as soon as it is written
- Applications of Python: Python used in many application domains.
 - Web and Internet Development
 - Scientific and Numeric
 - Education
 - Desktop GUIs
 - Software Development
- Business Applications

4.2 Testing Techniques and Test plans:-

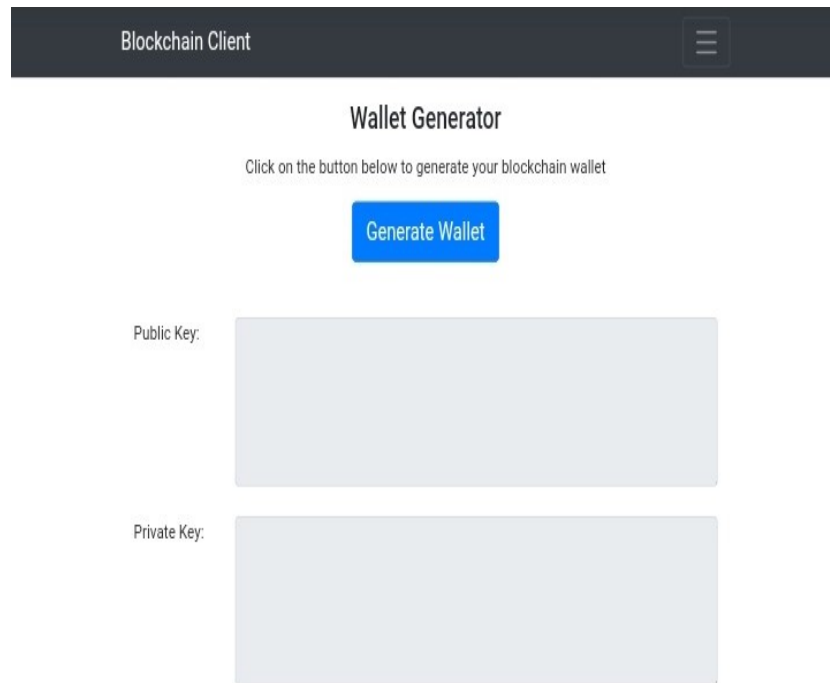
- We will test our system by generating wallets. If the generated wallet contains public and private key then test is successful.
- We will do the transaction using these public and private keys. If the Transaction is confirmed then test is successful.
- We will make the transaction. Then try to view the transaction without mining the block . If user cannot see the transaction then system is working fine. Because without mining the block and added the Transaction to the block then add these block to blockchain no user can see the transaction as Transaction are not confirmed fully.
- Try to mine the block to see that Transaction is added on the block or not . If Transaction added to the block then our test is successful.

5 Result and Discussion

We implement the algorithm of the system using Pycharm/Google Colaboratory using python language. We implement our web application on a temporary server generated by “Flask” an Python web framework. The result for implementing our web application on temporary server are given below :-

5.1 Snapshot of system with brief detail of each :-

- First open the server at which “Blockchain Client” is running. Then a new dashboard will open.



The image shows a web application interface for a Blockchain Client. At the top, there is a dark grey header bar with the text "Blockchain Client" on the left and a hamburger menu icon on the right. Below the header, the main content area has a title "Wallet Generator" in bold. Underneath the title is a small instruction: "Click on the button below to generate your blockchain wallet". A prominent blue button with the text "Generate Wallet" is centered below the instruction. Below the button, there are two input fields. The first is labeled "Public Key:" and the second is labeled "Private Key:". Both labels are in a small, light grey font. The input fields themselves are large, light grey rectangular boxes.

Figure 7: Blockchain Client Dashboard

- Then generate wallet by click on “Generate Wallet” button to generate public key and private key. Here public key use as addresses for sender and receiver. And private key is for validation of transactions. You have to generate two wallets for sender and receiver both.

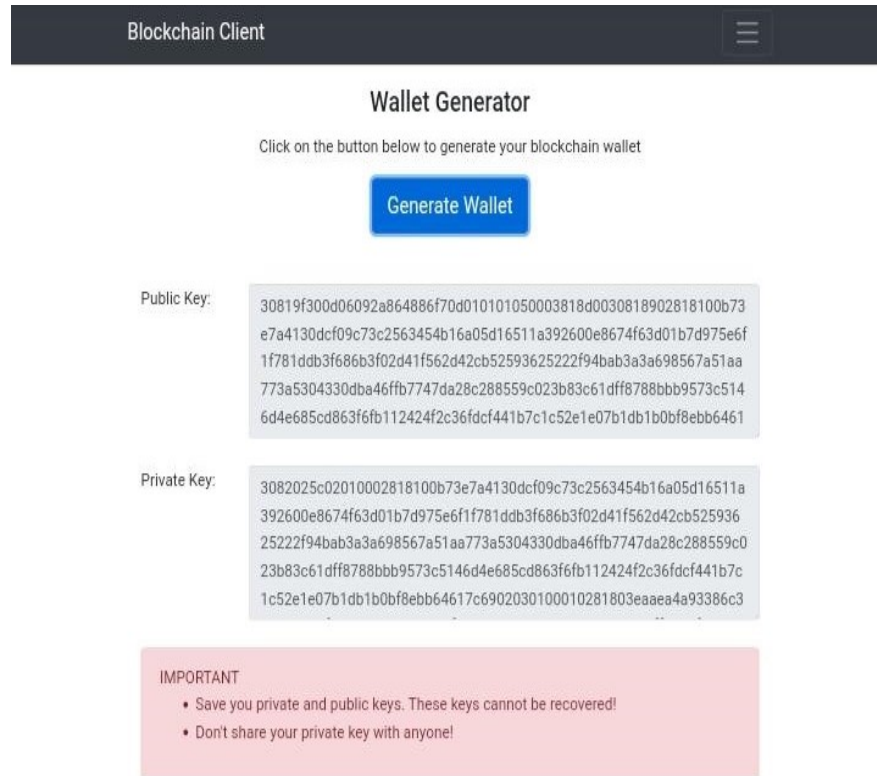


Figure 8: Generating wallets containing public and private key

- Then open the drop down menu in top right corner where you can find three option “Generate Wallet”, “Make Transaction” and “View Transaction”. Select the option “Make Transaction” . The new dashboard will open.



Figure 9: Drop down menu of dashboard

Blockchain Client

Send Coins

Enter transaction details and click on "Generate Transaction" button to generate your transaction

Sender Address:

Sender Private Key:

Recipient Address:

Amount to Send:

Generate Transaction

Figure 10: Dashboard for Transaction details form

- Fill the transaction detail. Here addresses of sender and receiver is the Public key generated by “Generate Wallets” dashboard.

Blockchain Client

Send Coins

Enter transaction details and click on "Generate Transaction" button to generate your transaction

Sender Address:

30819f300d06092a864886f70d010101050003818d0030818902818100b4

Sender Private Key:

588e9c258084b4b804657ea54becaf30ee9c02663f05394f0ee2664dff44e6

Recipient Address:

30819f300d06092a864886f70d010101050003818d0030818902818100ab

Amount to Send:

1000

Generate Transaction

Figure 11: Filled form of transactions details

- Now click on the “Generate Transaction” to generate the transaction. A new dialog box for confirmation transaction will open. Here you have to fill the “Blockchain Node Url”. In order to make or view transactions, you will need at least one blockchain node running.

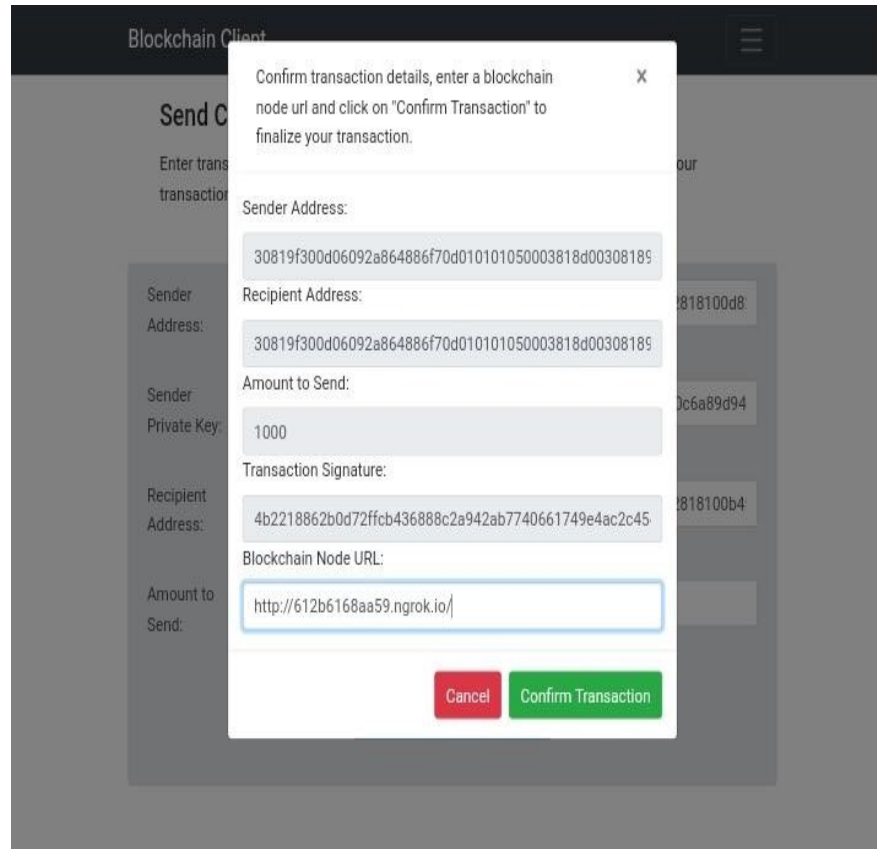


Figure 12: Confirm Transaction Dialogue Box

- Now click on “Confirm Transaction” button. The message will be prompt of successful transaction.

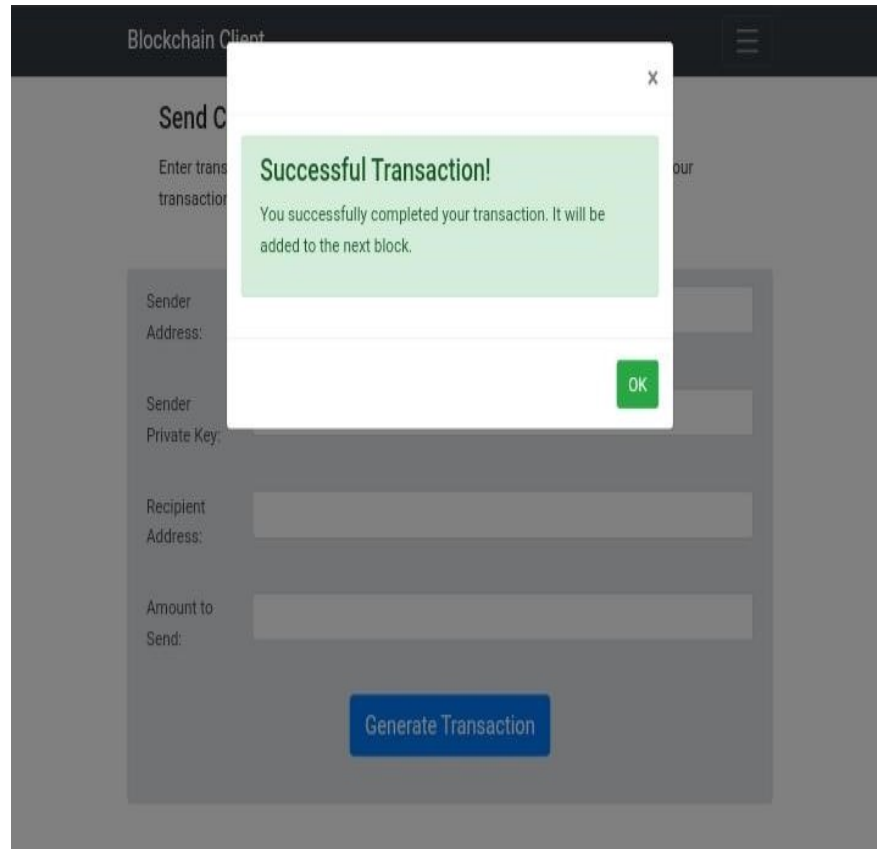


Figure 13: Showing Successful Transaction Message

- Now open the another server at which “Blockchain Frontend” is running. The new dashboard will open.

Blockchain Frontend

Add Blockchain nodes

Enter a list of Blockchain node URLs separated by comma and click on "Add" button to add them to the list of nodes

Node URLs:

Add Node

This node can retrieve Blockchain data from the following nodes:

- 612b6168aa59.ngrok.io

Figure 15: Adding Node in the Node URLs

- The click the “Add Node”. Button to add the node in the system. Then node will be add.

Blockchain Frontend

Transactions to be added to the next block

Show 10 entries

Search:

#	Recipient Address	Sender Address	Value
1	30819f300d06092a864886f7...	30819f300d06092a864886f7...	1000

Showing 1 to 1 of 1 entries

Previous

1

Next

Mine

Transactions on the Blockchain

Show 10 entries

Search:

#	Recipient Address	Sender Address	Value	Timestamp	Block
No data available in table					

Showing 0 to 0 of 0 entries

Previous

Next

Figure 16: Node added for retrieving

- Now select the option mine from the drop down menu. There you can see the transaction we have done on the “Transaction to be added in the block”.

Blockchain Frontend

Transactions to be added to the next block

Show 10 entries

Search:

#	Recipient Address	Sender Address	Value
1	30819f300d06092a864886f7...	30819f300d06092a864886f7...	1000

Showing 1 to 1 of 1 entries

Previous

1

Next

Mine

Transactions on the Blockchain

Show 10 entries

Search:

#	Recipient Address	Sender Address	Value	Timestamp	Block
No data available in table					

Showing 0 to 0 of 0 entries

Previous

Next

Figure 17: Showing Transaction to be added on the block

- Now click the “Mine” Button. Then you will see the transaction has been added in the blockchain.

Blockchain Frontend

Transactions to be added to the next block

Show 10 entries

Search:

#	Recipient Address	Sender Address	Value
No data available in table			

Showing 0 to 0 of 0 entries

Previous

Next

Mine

Transactions on the Blockchain

Show 10 entries

Search:

#	Recipient Address	Sender Address	Value	Timestamp	Block
1	30819f300d06092a864886f7...	30819f300d06092a864886f7...	1000	Jun 7, 2021, 12:21:34 PM	2
2	315dda69e70d405bbd8f5f9f...	THE BLOCKCHAIN	1	Jun 7, 2021, 12:21:34 PM	2

Showing 1 to 2 of 2 entries

Previous

1

Next

Figure 18: Transaction added on the Blockchain

- Now go back to the “Blockchain Client” and select the View Transactions on the drop-down menu. Add the node in the node URL.

31

Blockchain Client

View Transactions

Enter a blockchain node URL and click on "View Transactions" button to check all transactions

Node URL:

View Transactions

Figure 19: Adding node to the Node URL

- Now click on the “View Transactions” button. There you can see that your Transaction is fully completed and added to the blockchain.

Blockchain Client

View Transactions

Enter a blockchain node URL and click on "View Transactions" button to check all transactions

Node URL:

View Transactions

Show 10 entries

Search:

#	Recipient Address	Sender Address	Value	Timestamp	Block
1	30819f300d06092a864886f7...	30819f300d06092a864886f7...	1000	Jun 7, 2021, 12:21:34 PM	2
2	315dda69e70d405bbd8f5f9f...	THE BLOCKCHAIN	1	Jun 7, 2021, 12:21:34 PM	2

Showing 1 to 2 of 2 entries

Previous1Next

Figure 20: Transaction can be viewed by the client

6 Conclusion and Future Scope

6.1 Conclusion

This project is to provide users a peer-to-peer to system for transaction of digital currency without any central authority. This system provides users to generate wallets with private key and public key used for verification of the transactions.

6.2 Future Scope

This project helps us in future to implement Blockchain Technology in many areas where transaction is involved. Implementing this system on different platforms (mobile application and desktop application.

7 References/Bibliography

- <https://www.investopedia.com/terms/b/blockchain.asp>
- <https://www.euromoney.com/learning/blockchain-explained/what-is-blockchain#:~:text=Blockchain%20is%20a>
- https://www.tutorialspoint.com/python_blockchain/index.htm
- <https://www.tutorialspoint.com/blockchain/index.htm>