

Project 3 - Final Report

Ece Nur ŞEN - 150150104

Joshgun Rzabayli - 150160901

Main Objective

Main objective for the third project is to create our own cryptocurrency.

Duration

Duration for this project is in total three weeks. The deadline is 28th of December, 2020.

Planning

Main objectives are listed below in itemized form:

1. Cryptocurrency client
 - a. Public-private keys
 - b. Communicating with network
 - c. Receiving blocks from network
 - d. Creating and posting new transaction to network
 - e. Creating and posting new blocks from network
2. Verifications
 - a. Block verification
 - b. Transaction verification
 - c. Block fee

Overview

In this project, we created a cryptocurrency called “Happy Coin”. For this project, we used python as a programming language. In the end of the project, we were planning to provide a website, however we had some problems and couldn’t host the website at the same time with peer to peer network locally. We decided to use the Graphical User Interface of Python which is called Tkinter(Tk) [1]. Tkinter is the standard Python interface that is available on most Unix platforms, as well as on Windows systems. We developed a simple interface where users can use our Happy Coin network to make transactions and observe transactions. According to the fundamentals of blockchain, every client is a node and they are contributing to the “Happy Coin” network.

Communication

For communication, TCP-based peer-to-peer connection is established between nodes. Creating a P2P network is an extremely challenging process. To create a whole P2P network, a HappyCoin client must have multiple threads in order to communicate the whole network. For P2P, we used socket [2], threading [3] and sys [4] libraries in python.

When a HappyCoin Client is created/woke up, we are first checking whether the client is an old or new client. Old clients will have a file where they store their previously created public/private keys. If the client is old, we are fetching the key information from the related file, otherwise, a new public/private key is generated. After creating the client node, the node is connecting to the whole P2P network by connecting through all nodes in the network and it requests the block history from the network. Without downloading the whole network, a node cannot create, send transactions or blocks.

Blockchain structure

Our Blockchain network consists of 3 classes: Blockchain class, Block class and Transaction class.

Blockchain Class is the main part of the network which includes necessary methods such as node creation, adding new transactions, creating new blocks, verifying the chain, generating keys and etc. During the creation of the new Blockchain object, the first block is created as default and added to the block list kept in the Blockchain object. As soon as the user submits a new transaction, a new 64-bit transaction ID is created for the new transaction and added to the list of unconfirmed transactions kept in the blockchain object. In order to create a new block, there must be at least 4 unapproved transactions, for this purpose, there is a transaction selector in the Blockchain class and a method that creates a new block for the selected transactions. On top of that, there are also functions inside the class that check the validity of all blocks in the blockchain, show the total balance or balance of only approved transactions, and generate new blocks for the created transactions(mining).

The secrets [5] module of the Python is used to create a Private Key for every new node. The reason we use this library is because it generates cryptographically strong random numbers. The difference from the random module is the ability to generate numbers that can be used for security reasons. We generated a 256 bits random number for the Private Key value. A Python module named Bitcoin Keygen[6] was used to create a new public key from the created 256-bit Private key and a to create a new address from this public key.

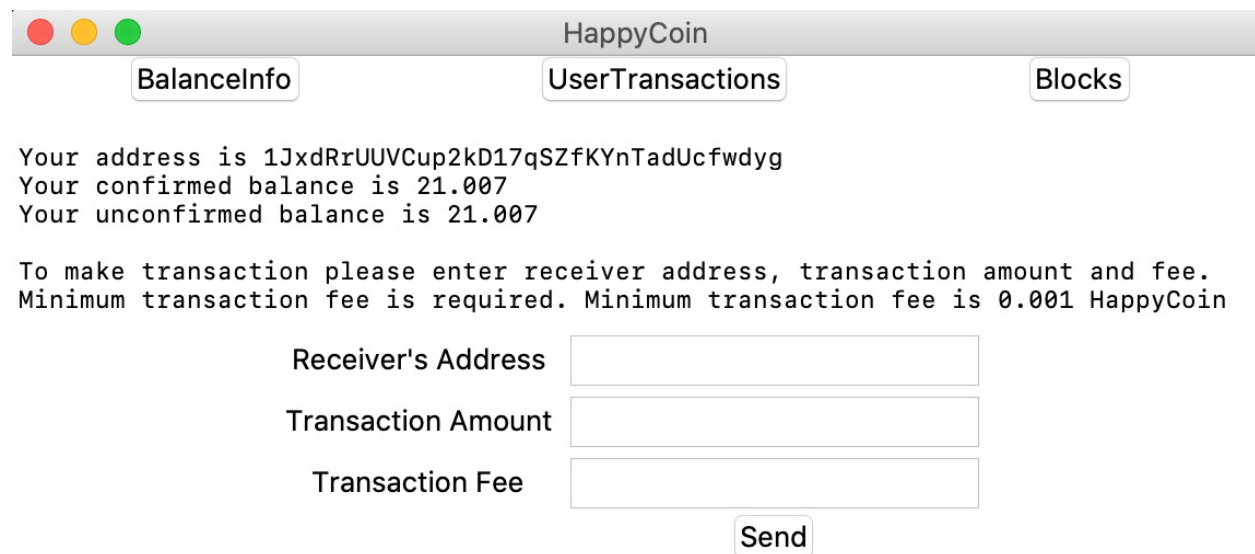
Block class is used to create new blocks in the chain and add them to our Blockchain. This class has some functions such as constructor of the block class and hash code generator method which is used to create new hash code for every new block. The Python Hashlib [7] library is used to generate hash codes. Moreover, there are 2 static methods that are created for converting a Block object to Python Dictionary format or vice-versa.

The Transaction class is used to create new transactions with specified sender address, receiver address, transaction amount and transaction fee(default is 0.001). The transaction ID for all the Transaction objects is generated by the secrets module of the Python as mentioned

above. This ID value is 64 bits. On top of that, there are 2 static methods that are created for converting a Transaction object to Python Dictionary format or vice-versa.

Application Flow

Firstly, the user sees the main page as in Figure 1. Navigation bar is located on the top of the page and has buttons named: BalanceInfo, UserTransactions, Blocks. There is information about the address of the node, confirmed balance and unconfirmed balance. Moreover, there is a form to generate a new transaction with the receiver's address, transaction amount and transaction fee attributes.



HappyCoin

BalanceInfo UserTransactions Blocks

Your address is 1JxdRrUUVcUp2kD17qSZfKYnTadUcfwdyg
Your confirmed balance is 21.007
Your unconfirmed balance is 21.007

To make transaction please enter receiver address, transaction amount and fee.
Minimum transaction fee is required. Minimum transaction fee is 0.001 HappyCoin

Receiver's Address

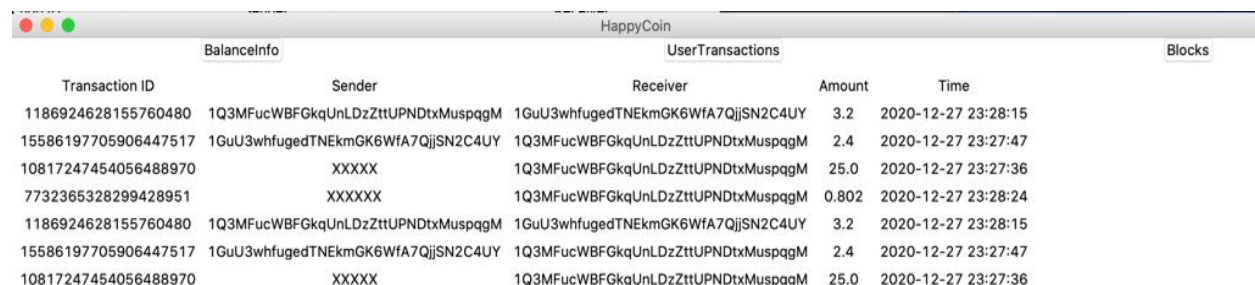
Transaction Amount

Transaction Fee

Send

Figure 1: Main Page

All transactions of the specified node are listed in the transactions page as shown in Figure 2.



Transaction ID	Sender	Receiver	Amount	Time
1186924628155760480	1Q3MFucWBFgkqUnLDzZttUPNDtxMuspqgM	1GuU3whfugedTNEkmGK6WfA7QjJSN2C4UY	3.2	2020-12-27 23:28:15
15586197705906447517	1GuU3whfugedTNEkmGK6WfA7QjJSN2C4UY	1Q3MFucWBFgkqUnLDzZttUPNDtxMuspqgM	2.4	2020-12-27 23:27:47
10817247454056488970	XXXXX	1Q3MFucWBFgkqUnLDzZttUPNDtxMuspqgM	25.0	2020-12-27 23:27:36
7732365328299428951	XXXXXX	1Q3MFucWBFgkqUnLDzZttUPNDtxMuspqgM	0.802	2020-12-27 23:28:24
1186924628155760480	1Q3MFucWBFgkqUnLDzZttUPNDtxMuspqgM	1GuU3whfugedTNEkmGK6WfA7QjJSN2C4UY	3.2	2020-12-27 23:28:15
15586197705906447517	1GuU3whfugedTNEkmGK6WfA7QjJSN2C4UY	1Q3MFucWBFgkqUnLDzZttUPNDtxMuspqgM	2.4	2020-12-27 23:27:47
10817247454056488970	XXXXX	1Q3MFucWBFgkqUnLDzZttUPNDtxMuspqgM	25.0	2020-12-27 23:27:36

Figure 2: Transactions Page

The Blocks Page is used to display all blocks which are created by the node's address as in Figure 3.

HappyCoin			
BalanceInfo	UserTransactions		Blocks
Block Hash	Block Time	Number of Transaction	No. Confirmation
00000277765219f08664b017e2793a24e2d90f2111dc1b9bc9f7fbd05ae901f5	2020-12-27 22:54:03	5	1
00000db680d18925a7cadea7719637e775f70fe4715c82d1d03d20189fed5f09	2020-12-27 22:51:09	5	2
000004ee464cc8e0f6cb133d99b7a9c5c36d2a35c1918f0728bfd8473a87d99	2020-12-27 22:48:42	5	3

Figure 3: Blocks Page

References

- [1] Tkinter - Python interface to Tcl/Tk¶. (n.d.). Retrieved December 27, 2020, from <https://docs.python.org/3/library/tkinter.html>
- [2] Socket - Low-level networking interface¶. (n.d.). Retrieved December 20, 2020, from <https://docs.python.org/3/library/socket.html>
- [3] Threading - Thread-based parallelism¶. (n.d.). Retrieved December 20, 2020, from <https://docs.python.org/3.9/library/threading.html?highlight=threading>
- [4] Sys - System-specific parameters and functions¶. (n.d.). Retrieved December 20, 2020, from <https://docs.python.org/3/library/sys.html>
- [5] Secrets - Generate secure random numbers for managing secrets¶. (n.d.). Retrieved December 27, 2020, from <https://docs.python.org/3/library/secrets.html>
- [6] Bitcoin-keygen. (n.d.). Retrieved December 27, 2020, from <https://pypi.org/project/bitcoin-keygen/>
- [7] Hashlib - Secure hashes and message digests¶. (n.d.). Retrieved December 20, 2020, from <https://docs.python.org/3/library/hashlib.html>