Journal of Computer Science and Information Technology
December 2021, Vol. 9, No. 2, pp. 4-12
ISSN: 2334-2366 (Print), 2334-2374 (Online)

Published by American Research Institute for Policy Development
DOI: 10.15640/jcsit.v9n2a2
URL: https://doi.org/10.15640/jcsit.v9n2a2

Voting Application Implementing Blockchain

Antoine Whitaker[1], Jean Muhammad[2], Moayed Daneshyari[3], Chutima Boonthum-Denecke[2]

Abstract

Cryptocurrencies are becoming increasingly popular due to the spike in the price of bitcoin. Bitcoin uses a technology called blockchain in order to make secure and anonymous transactions. This paper will discuss how to implement a voting application in Node JS in order to prevent malicious activity during the election periods. This is accomplished through implementation of blockchain technology.

Keywords: Blockchain, Bitcoin, Cryptocurrency, Blockchain mining, Peer-to-peer, Node JavaScript, Markle Tree

1. Introduction

Blockchain is a growing technology used in the cryptocurrency market to record chronological and public transactions made in bitcoin and other cryptocurrencies (Crosby et al., 2015; D'Aliessi, 2016). A blockchain is basically a distribute database of records, or public ledger of every digital event/transaction that has been performed among different parties. Each event in the public ledger is confirmed by an agreement of most of participants in the whole system. The data entered into this system cannot be erased (Crosby, Pradan, Sanjeev, and Vignesh , 2016). The blockchain comprises a confirmed record of every single transaction that has ever been made (D'Aliessi, 2016). This technology allows people to exchange currency without a central authority such as a bank, on a global scale that works based on the trust to the authority of the whole (Crosby, Pradan, Sanjeev, and Vignesh , 2016).

When two parties would like to perform transactions, this can be done via (1) trusting each other, (2) creating a contract signed by both parties, (3) using a central location like a bank to keep the money in order to perform the transaction. These methods are not optimal and can fail, while the blockchain technology provides security, anonymity, and data integrity without any third party organization in control of the transactions (Crosby, Pradan, Sanjeev, and Vignesh, 2016). This technology allows the creation of decentralized currencies, self-performing digital contracts and intelligent assets controlled over web, thus permits the expansion of new governance systems with more participant with autonomous organizations with no human intervention (D'Aliessi, 2016). The development of blockchain technology can potentially affect future of many research fields from machine learning and data analytics to applications such as medical managements.

The disruptive innovation will potentially enable us to redesign our interactions in business, politics and society at large. Although research interest in the theory of the subject is growing, a complete analysis of blockchain applications from a political perspective would be needed. In this regard, in this paper, we provide an implementation of a voting application using blockchain technology.

[1] Microsoft
[2] Department of Computer Science, Hampton University
[3] Department of Computer Science, California State University East Bay (Corresponding author: moayed.daneshyari@csueastbay.edu)

2. Blockchain Technology

Blockchain technology has become increasingly popular with the rise in the price of bitcoin, a well-known cryptocurrency (D'Aliessi, 2016). This currency was created to exchange products and services like a country's currency is used. You may think of a single bitcoin as a dollar. It has no value of its own, instead the value is introduced when people agree to trade goods and services to bring more of the currency under their control, and hope others are doing the same. A ledger is used by the blockchain to keep track of bitcoin transactions. The ledger is not stored in a central location. It is distributed amongst a private network of computers called miners. These miners are responsible for storing data and executing computations. Every miner represents a node in the blockchain network and has its own copy of the ledger. In Figure 1 below David is sending Sandra 5 BTC. He broadcasts a message to the network that contains the number of bitcoins needed for the transactions. The number of bitcoins in Sandra's side should increase by that same amount. Every node in the network will receive message and update their copy pf the ledger and the account balance for both parties accordingly (D'Aliessi, 2016).

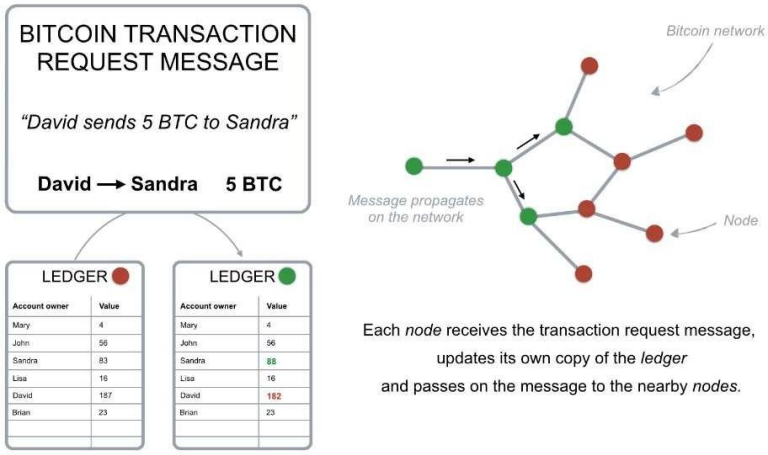

Figure 1: Transaction request message (D'Aliessi, 2016)

The ledger being manipulated by a network of computers rather than a central location provides some things to consider:

- In a banking system, people can only see their account information, blockchain allows everyone to see every person's transaction.
- For the most part, you can trust your bank, while in the bitcoin network, there is no helpdesk or a person to sue, should something go wrong.
- Blockchain is designed so security and reliability are obtained by mathematical functions and code, so there is no need to trust to a person.

2.1 Wallet

Blockchain is defined as a system that allows groups of connected computers to maintain a single updated and secure ledger. To execute these transactions on the blockchain, you need a wallet, which is a program that allows you to store and exchange your bitcoins. You are the only person that should be allowed to spend your bitcoins, so each wallet is protected by a special cryptographic method that uses a unique pair public and private keys (Aziz, 2018).

This encryption works by encrypting a message and only the owner has the private key to decrypt and read that message. This also works in reverse, if a message is encrypted with a private key, only the paired public key can decrypt it. When Bob sends bitcoins to others, he needs to broadcast a message encrypted with the private key of his wallet. This makes Bob the only one who has the private key necessary to unlock his wallet to spend his bitcoins.

Each node in the network can check that the request came from Bob by decrypting the message with the public key associated with his wallet (Aziz, 2018). Aziz said that a digital signature is generated when a transaction request is encrypted with the private key in your wallet, so blockchain computers can verify the source and authenticate the transaction. The digital signature is a string of text resulting from transaction request and your private key so they cannot be reused in other transactions. Changing a single character will result in a completely different signature. This prevents attackers from changing your transaction request or alters the number of bitcoins when sending transactions. In order for bitcoins to be sent, you need to prove you own the private key of the specific wallet because you need the key to encrypt your transaction request message. You never have to reveal the private key because you broadcast the message after it has been encrypted.

2.2 Wallet Balance

Every node in the blockchain has a copy of the ledger. The system does not keep track of the account balance for anyone. It only records each transaction that is verified and approved. The ledger stores records of every transaction broadcasted within the bitcoin network and no actual account balances. Your wallet balamce is determined by analyzing and verifying all the transactions that ever took place on the entire network connected to your wallet. The verification of the balance can be done thanks to links to previous transactions. To speed up the verification process a special record of unspent transactions is kept by the network nodes. This security check prevents the double spending problem (Brikman, 2014).

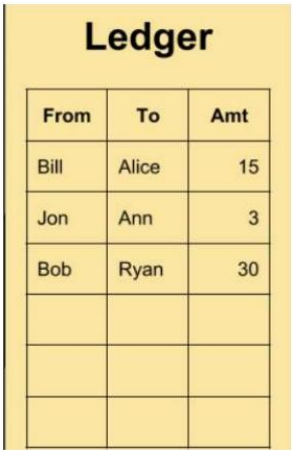

Figure 2: Bitcoin ledger example (Brikman, 2014)

Owning bitcoins means your bitcoin wallet address is being referenced by transactions that have not been used as input. The bitcoin transaction code is all opensource. All you need is a laptop and an internet connection to start transactions. One shall be aware of mistakes in the code used to broadcast a transaction request message because it is possible to have the associated bitcoins involved in a transaction will be permanently lost. It should always be noted that there is no helpdesk or a customer service to call should something go wrong. To minimize the risk, one shall use the Bitcoin Core (official version of bitcoin wallet software), and store the wallet's password or private keys in a secure repository.

3. Security

There are 2^{160} bitcoin addresses (Bork, 2018). This is a protection mechanism to prevent the bitcoin network from being attacked while also allowing anyone to own a wallet. People can make anonymous transmit and receive transaction and only reveal their public key. However, they can also use said public key constantly, may result in someone connecting all the transaction to the same owner. The bitcoin network allows use to create as many wallets as you want. Each wallet had different public and private keys. This approach prevents anyone from knowing you have multiple wallets, unless you transfer all the bitcoins to a specific wallet.

There us a serious security hole with the setup discussed previously (Bork, 2018). Transactions are passed from node to node so the speed at which a multiple transaction reach each node is different. Attackers can send a transaction, wait for the counterpart to ship a product, and then send a reverse transaction back to his account. This case is handled by having some nodes receive the second transaction before the first and therefore consider the initial payment transaction invalid because the transaction is already spent. We also know which transaction was sent first because of the timestamp. However, it is not secure to order the transactions by timestamp, because it could be counterfeited. So, we cannot determine what happened first. In the event of a dispute among the network of nodes regarding to order transactions and prevent fraud.

Transactions are ordered by grouping them into blocks. Each block contains a specific number of transactions and a link to the previous block. This is how the block are placed in a chain-like format and organized into a time related chain. This idea was where the name blockchain originated. Transaction in the same block are considered to have happened at the same time and any transactions not present in a block are considered unconfirmed. Every node is capable of grouping transactions into a block and broadcasting it to the network as a suggestion for which block should be added next.

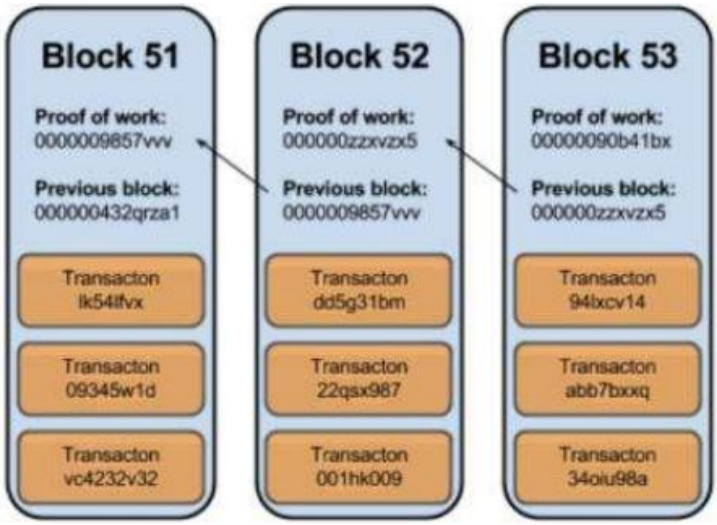

Figure 3: Example of blockchain (Bork, 2018)

In order for a node to add a block to the chain, each block must contain the answer to a complex mathematical problem created using an irreversible cryptograph hash function (D'Aliessi, 2016). The only way to solve the problem is to guess random numbers that combine with the previous block content and generate a defined result. A typical computer can take a year to guess the right number and solve the mathematical problem. This process is sped up due to the large number of computers in the network. On average a new block is added to the chain every 10 minutes. The node that solves the problem gains the right to add its block to the chain and broadcast a message to the network. If two nodes solve the problem at the same time, both blocks are broadcasted, and each node builds on the block that is receives first. However, the blockchain system requires each node to build immediately on the longest blockchain available. If ambiguity exists about the last node added, as soon as the next block is solved, every node will adopt the longest chain. It is an extremely low possibility that twi nodes solve the problem simultaneously, so this case is almost impossible to occur.

This situation is rare but if it does happen, it could possibly lead to a double spending problem. Susan can send money to Jack, and Jack ships the product to Susan. Since the nodes always adopt the longer tail as the confirmed transactions, Susan could generate a longer tail that contains a reverse transaction. She could manage to generate the same input references, leaving Jack without money or his product. This kind of fraud is prevented by the mathematical problem needed for the new blocks. A part of the problem contains a reference to the previous block. So, it is exceedingly difficult to precompute a series of blocks due to the high number of random guesses needed to solve a block. Susan has to race against the entire network is order to

accomplish this. Even id se does manage to calculate solve the problem, it is very unlikely for her to solve multiple in a row. A person would need to control 50% of the computing power of the whole network to have a 25% chance of solving two blocks in a row (D'Aliessi, 2016).

Transactions are protected by the mathematical race, preventing attackers from unwanted behavior. Also, transactions grow more secure with time. The blocks added to the chain one hour ago are more secure than the ones added 10 minutes ago. This is because the block added an hour ago has most likely been processed and is irreversible.

4. Mining

Rewards are given to those who are able to solve the math problem. Mining is running the bitcoin blockchain software in order to obtain bitcoin rewards. This process is similar to mining gold. These rewards are the main drive for people to operate the necessary computing power for processing transactions. It takes a while for a computer to solve this math problem, so the work is divided amongst the nodes in the network to speed up the process. Working in groups, called mining pools, to guess the right solution and dividing the reward among the members us a good approach to take.

Some mining pools are so large that they account for 20% of the computing power in the network. This poses a problem seen earlier in the double spending problem. If one of the pools gains 50
50% of the computing network computing, the longer the chain the more secure the transactions become (D'Aliessi, 2016). Some of the miners limited the number of members in order to safeguard the overall security of the network. Since computers are getting faster and the number of nodes is increasing, the blockchain system is able to increase the difficulty of the mathematical problem to slow down the process in order to maintain the 10-minute average. Every four years, the reward for mining blocks is cut in half, so mining gets less interesting over time. In order to keep nodes operating, small reward fees can be attached to each transaction. Transactions that offer larger rewards are processed faster. When you send a transaction, you can decide if you want to process it fast, which is more expensive, or slower, takes more time but cheaper. In the bitcoin network, transactions fees are very small compared to bank charges, and they are not associated with the transaction amount.

Pros:

- No third party limiting or holding your funds. You have complete control.
- Low transaction cost worldwide so micropayments are possible
- Funds can be transferred in minutes and secured after a few hours
- Anyone can verify transactions made on the blockchain
- Use blockchain technology to decentralize applications

Cons:

- Transactions can be sent and received anonymously, allowing for illegal activity
- Not easy to trade bitcoins for goods and services
- Very volatile

Now let us discuss the reasoning on why we choose to implement blockchain in a voting application.

5. Applications of Blockchain

During the US Presidential Election in November 2016, there was a lot of speculation about how the results were tampered with (Ward, 2017). Russian hackers were able to breach the voting system in 39 of 50 states. The hack in Illinois provided the most information when General Council for Illinois Board of Election noticed unauthorized data leaving the network. This data contained the information of around 15 million people and included names, birthdays, genders, and partial Social Security numbers. The intruder's goal was to delete and alter the data they were able to take.

A hack on the state level would not accomplish this task because the country uploads the voter data to the states (Ward, 2017). Even if the data was deleted at the state level there was still a backup plan for replacing all the names. In an unknown state, Russians were able to get access to a database that had insight to the financial connections voters had to their candidates. Our application hopes to use blockchain as way to solve the security issues if the vulnerable US election system.

6. Implementing Blockchain in Node JavaScript

7.

We choose to implement blockchain in Node JavaScript due to JS allowing input and output communication methods. Node initiates the event loop at the start, process input, and starts the order of operations. Node JS also has the following advantages:

- Fast and scalable web apps
- For server-side applications (flow is determined by events)
- Scale on individual process basis spreading load to multi code servers
- Multiple modules (NPM, Grunt) and supportive community
- Good for real-time apps
- Lonest running programming language
- Most of developers are familiar with
- Sponsored by big corporations such as Microsoft, PayPal, Joylent, and Wal-Mart

7.1 Node JS Backend

Our application has three classes, app.js, brewNode.js, and brewChain.js. We start discussing the app.js class. This class is like the driver for a java program. This is the first class that is executed, and it uses the other classes for support. This class communicates with the frontend and backend passing data in both directions. It receives commands to create a block with the specified data and passes it to the createBlock() function in to brewNode.js. The data is then passed to the createBlock() method in the brewChain class where it is turned into a block as show in Figure 4. In this function, several important things would happen, The Merkle root is created from the vote being passed in.

```
function createBlock(data){
    let newBlock = {
        timestamp: new Date().getTime()
      , data: data
      , index: currentBlock.index+1
      , previousHash: currentBlock.hash
      , nonce: 0
      , hash: 999999999999999
      , merkleRoot: 0
    };

    newBlock.merkleRoot = createTree(newBlock.data);

    //Getting a hash for the current block for mining
    newBlock.hash = createHash(newBlock.merkleRoot, newBlock.timestamp,
        JSON.stringify(newBlock.data), newBlock.index,
            newBlock.previousHash, newBlock.nonce);

    newBlock = mineBlock(newBlock);

    return newBlock;
}
```

Figure 4: Display of the block structure

The Merkle root is created from a Merkle tree, a binary tree of hash values. Merkle tree starts with the data which can be private key, file, etc.

The leaf nodes represent the data's hash value. Multiple leaf nodes are combined and rehashed to create nodes. These nodes are the appended and rehashed into the root. This root is returned from the tree and then stored in a block as the merkleRoot variable. Parent nodes are the combination of its children and then reshaped. This process is illustrated in Figure 5 below.

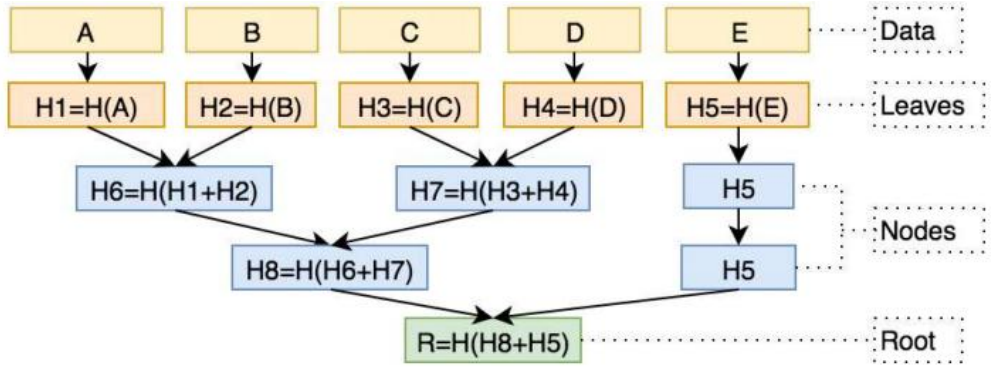

Figure 5: Merkle Tree (Mota, 2017)

After the Merkle root is added to the block, we calculated the hash of the current block (Mota, 2017). The hash is calculated by using the crypto sha256 encryption library. We pass all the values associated with a block in order to compute a hash. After an initial hash is generated, we mine for a hash that has a specific number of zeros. In our implementation we use three zeros to start our hash so we can demo and test it quickly. The number of zeros we look for is based on the difficulty. If the difficulty is three then we mine for a hash with three zeros. With the difficulty set to 3, when the program is executed, it finishes almost instantly. If we double the difficulty, then it takes to create blocks in the chain.

During the mining process, we pass in the same values each time, so the exact same hash value is returned. Each unsuccessful iteration of the mining function we increment a variable called a nonce so a different hash can be generated. This can be seen in Figure 6. Once the mining is complete that we return the block to the brewNode, which returns the block to the app.js class to be logged in the terminal and sent to the frontend to be displayed. Also, a request is sent to the other peers that a new block has been added to the chain. The peers get the updated and go through a series of validations before the chain is replaced in the nodes without the newly added block.

```
function mineBlock(newBlock){
    //Creating a string of 0 to show computing power
    while(newBlock.hash.substring(0, difficulty) !== Array(difficulty + 1).join("0")){
        newBlock.nonce++;
        newBlock.hash = createHash(newBlock.timestamp, JSON.stringify(newBlock.data),
            newBlock.index, newBlock.previousHash, newBlock.nonce);;
    }

    console.log("Block mined: " + newBlock.hash);
    return newBlock;
};
```

Figure 6: Our mining function.

7.2 Peer-to-Peer

The peers in our project are represented by terminal or command line windows. We are using an HTTP server and web interface to interact with brew activities, log, and other data. Each BrewNode server will have

a separate chain that it will manage, a web socket server, and an HTTP server to control it. When the init function is called, our server sets up a listening socket server. A handler is needed for incoming messages to connect peers and disconnect peers from our collection of sockets. An HTTP server needs to be added in order to send commands to our server, this will also be where the UI interface is. Each node will be responsible for the following:

- Create a new block
- Request the latest block when a new node is added
- If the node has no blocks then replace the entire chain
- When a new block is received with a larger index then the block in this nodes chain determine if it is valid and up next in the chain, if so then add it, otherwise transactions are missed, replace the chain.

This is accomplished by making sure whenever a node generates a block, it will send it to all connected node. This can be tested by creating two nodes and connect them. Refer to Figure 9 below for assistance and running the following URL:

http://localhost:3004/addNode/18078

Figure 7: Image of the output wen you initially run app.js (Beck, 2017)

Running http://localhost:3004/spawnBrew/barry to create a block and add it to both chains.

Figure 8: Showing a block added to the chain (Beck, 2017)

7.3 Angular JavaScript Frontend

Angular uses HTML to define the app's user interface. HTML is easier to recognize then JavaScript because HTML is less likely to break. You also have the added benefit of having more developers tough the code when using HTML. HTML has special attributes to determine what gets loaded, but not how. This approach greatly simplifies application development is a "what you see is what you get" kind of way. Little time is spent on how a program flows and what gets loaded first, you just defined what is needed an Angular will handle the dependencies. In short, Angular allows you to write less code.

7.4 Frontend

The Connection between frontend and backend mainly established using AJAX requests. Backend is developed using Mainly NodeJS and frontend developed by using AngularJS, JavaScript, and HTML.

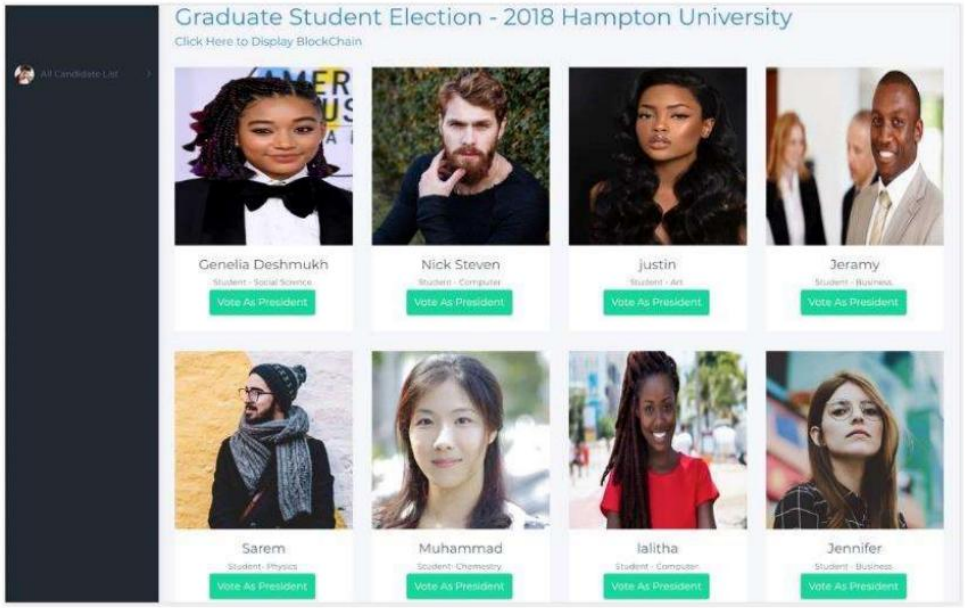

Figure 9: User Interface for application

8. Conclusion

We implemented blockchain in Node JS and connected our user interface in Angular. The purpose of our application is to prevent attackers from doing malicious things during elections and making it fair for all candidates involved. Blockchain is the great solution to prevent voting fraud. It also allows for citizens to see the live voting taking place and count the votes themselves. There is no way for hackers to change the results of the used ballots.

References

Aziz. (2018, November 5). Guide to cryptocurrency wallets: Why do you need wallets? Retrieved from https://masterthecrypto.com/guide-to-cryptocurrency-wallets/

Beck, D. (2017, December 10). NodeJS blockchain implementation: BrewChain: Chain websockets HTTP server. Retrieved from http://www.darrenbeck.com.uk/blockchain/nodejs/nodejscrypto/

Bitcoin mining. (2014, October 14). Everything you need to know about Bitcoin mining. Retrieved from https://www.bitcoingmining.com

Bitcoin. (2009) How does Bitcoin work? Retrieved from https://bitcoinf.org/en/how-it-works

Bork, L. (2018). Blockchain from scratch. Retrieved from https://medium.com/cwi-software/blockchain-from-sratch-pt-1-c7b80622b7f

Brikman, Y. (2014). Bitcoin by analogy. Retrieved from https://www.ybrikman.com/writing/2014/04/24/bitcoin-by-analogy/

Crosby, M., Nachiappan, Pattanyak, P., Verma, S., Kalyanaraman, V. (2015). Blockchain technology: Beyond bitcoin. The University of California Berkeley: Technical Report.

D'Aliessi, M. (2016). How does the blockchain work? – Member feature stories – Medium. Retrieved from https://medium.com/s/story/how-does-the-blockchain-work-98c8cd01d2ae

Mota, M. (2017, December 9). Miguemota/merkletreejs. Retrieved from https://github.com/miguelmota/merkletreejs

Ward, A. (2017, June 13). Russia hacked voting system in 39 states before the 2016 presidential election. Retrieved from https://www.vox.com/world/2017/6/13/15791744/russia-election-39-states-hack-putin-trump-sessions