MyBitcoin

I. MyBitcoin: A decentralized digital asset made for the masses 2
A. Abstract 2
B. Introduction 2
C. Blockchain 2
II. Key concepts 3
III. Block Process 4
A. Block structure 4
IV. Transactions 5
A. Transaction Flow 6
B. Transaction structure 6
V. Timestamp Server 7
VI. Proof of Work 7
A. Code Snippet: 7
B. Transaction Confirmation 8
VII. Network 8
VIII. Incentive / miner reward 9
A. Code Snippet: 9
IX. Reclaiming storage 9
X. Splitting of Value 10
A. Bits 10
B. How Is Bits Different From Other Digital Denominations? 10
C. How Many Dollars Is 1 Bits? 10
D. How Can I Buy Bits? 10
XI. Privacy / Encryption 11
A. Code Snippet: 11
XII. Setup and Operation 12

I. MyBitcoin: A decentralized digital asset made for the masses

A. Abstract

Mainstream Cryptocurrencies are gaining a foothold after a decade long hiatus. Even governments are taking cognizance of the power of Blockchain and going all out either to include cryptocurrencies or to block them to safeguard their hold on financial status of their citizens. We at Team MyBitcoin have come up with a novel approach to bring cryptocurrencies closer to everyday user. A lightweight, mobile device only, Proof of Work / Proof of Stake based, holding limited cryptocurrency with transaction based fee to make it pocket friendly for the user. A public Blockchain ensures a fair P2P (peer to peer) transaction which can be verified all the times.

B. Introduction

Modern banking and financial instruments are centralised systems which tend to control, predict and push the user in particular patterns. This was the main reason for the rise of Cryptocurrencies rise to prominence. These peer-to-peer systems of transactions enabled users to control how they spend money.

But, as of late large Banks, financial institutions with deep pockets and miners who threaten the cryptocurrency operations by pooling too much of the mining power (~51 percent) and absurd transaction fees have made cryptocurrencies a less popular option as financial instruments.

MyBitcoin is designed to overcome all these transactions. Mobile device only operations save the costs of mining to users, limiting of pooling of mining power reduces the threat of centralisation, high level of encryption ensures complete privacy. These features along with non-traceability feature makes MyBitcoin an ideal financial instrument to keep transactions discreet.

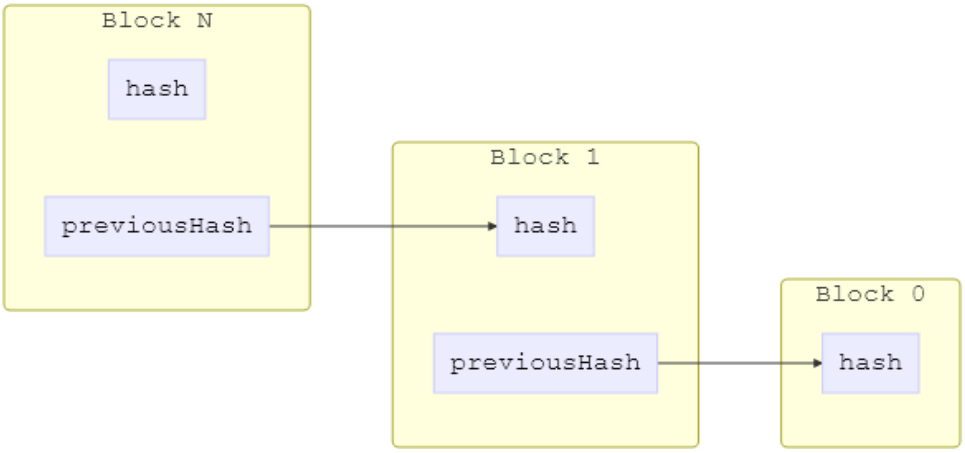
C. Blockchain

The blockchain holds two pieces of information, the block list (a linked list), and the transaction list (a hash map).

It's responsible for:

- Verification of arriving blocks
- Verification of arriving transactions
- Synchronization of the transaction list
- Synchronization of the block list

The blockchain is a linked list where the hash of the next block is calculated based on the hash of the previous block plus the data inside the block itself.



II. Key concepts

- Components
  - HTTP Server
  - Node
  - Blockchain
  - Operator
  - Miner
- HTTP API interface to control everything
- Synchronization of blockchain and transactions
- Simple proof-of-work
- Addresses creation using a deterministic approach [**EdDSA**](https://en.wikipedia.org/wiki/EdDSA)
- Data is persisted to a folder

MyBitcoin uses websocket for P2P communication, but it was dropped to simplify the understanding of message exchange. It is relying only on REST communication.

HTTP Server Provides an API to manage the blockchain, wallets, addresses, transaction creation, mining request and peer connectivity.

III. Block Process

A block is added to the block list if:

- The block is the last one (previous index + 1);
- The previous block is correct (previous hash == block.previousHash);
- The hash is correct (calculated block hash == block.hash);
- The difficulty level of the proof-of-work challenge is correct (difficulty at blockchain index _n_ < block difficulty);
- All transactions inside the block are valid;
- The sum of output transactions are equal the sum of input transactions + 50 coins representing the reward for the block miner;
- Check if there is a double spending in that block
- 8. There is only 1 fee transaction and 1 reward transaction.

A. Block structure

```
// Block
{
    "index": 0, // (first block: 0)
    "previousHash": "0",
    "timestamp": 1465154705,
    "nonce": 0,
    "transactions": [
        {
            "id": "63ec3ac02f...8d5ebc6dba",
            "hash": "563b8aa350...3eecfbd26b",
            "type": "regular",
            "data": {
                "inputs": [],
                "outputs": []
            }
```

```
            }
        ],
        "hash": "c4e0b8df46...199754d1ed"
    }
```


IV. Transactions

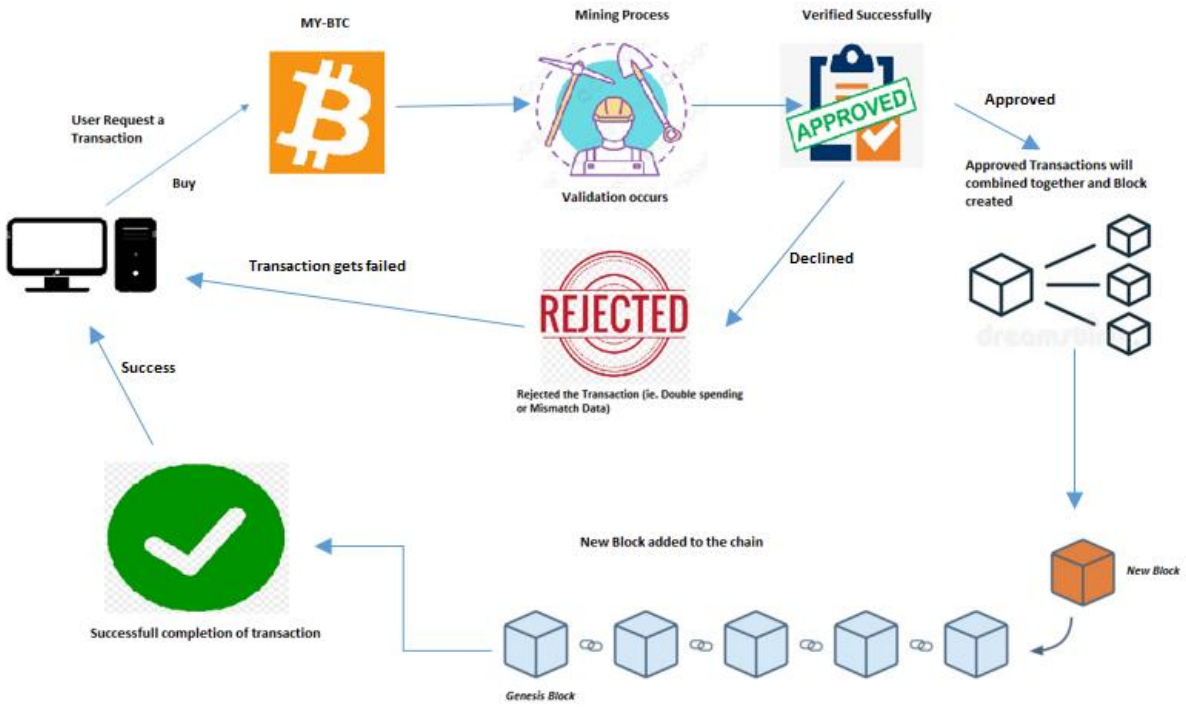
A simple PoW / PoS ensures a fluid transactions through the blockchain. A unique Digital signature which passes through SHA-256 ensures highest level of encryption. A lightweight, size limited block ensures faster transaction speeds. The transaction passes through a load balanced node network to go through the shortest path to ensure lightning transaction speeds.

To prevent the problem of double spending, a single history of transactions is published on a public blockchain in the form of hashes.

A transaction inside a block is valid if:

- The transaction hash is correct (calculated transaction hash == transaction.hash);
- The signature of all input transactions are correct (transaction data signature can be verified with the public key of the address);
- The sum of input transactions are greater than output transactions, it needs to leave some room for the transaction fee;
- The transaction isn't already in the blockchain
- All input transactions are unspent in the blockchain.

A. Transaction Flow



B. Transaction structure

A transaction contains a list of inputs and outputs representing a transfer of coins between the coin owner and an address. The input list contains a list of existing unspent output transactions and it is signed by the address owner. The output list contains amounts to other addresses, including or not a change to the owner address.

```
// Transaction
{
    "id": "84286bba8d...7477efdae1",
    "hash": "f697d4ae63...c1e85f0ac3",
    "type": "regular",
    "data": {
      "inputs": [ // Transaction inputs
          {
            "transaction": "9e765ad30c...e908b32f0c",
            "index": "0",
            "amount": 5000000000,
```

```
                "address": "dda3ce5aa5...b409bf3fdc",
                "signature": "27d911cac0...6486adbf05"
            }
        ],
        "outputs": [ // Transaction outputs
            {
                "amount": 10000,
                "address": "4f8293356d...b53e8c5b25"
            },
            {
                "amount": 4999989999,
                "address": "dda3ce5aa5...b409bf3fdc"
            }
        ]
    }
}
```

V. Timestamp Server

The time stamp works in tandem with the Transaction server for a call back function and as an added security feature to ensure that the transaction is stored appropriately on the public Blockchain.

VI. Proof of Work

The proof-of-work is done by calculating the 14 first hex values for a given transaction hash and increases the nonce until it reaches the minimal difficulty level required. The difficulty increases by an exponential value (power of 5) every 5 blocks created. Around the 70th block created it starts to spend around 50 seconds to generate a new block with this configuration. All these values can be tweaked.

A. Code Snippet:

```
const difficulty = this.blockchain.getDifficulty();
do {
    block.timestamp = new Date().getTime() / 1000;
```

```
            block.nonce++;
            block.hash = block.toHash();
            blockDifficulty = block.getDifficulty();
           }
        while (blockDifficulty >= difficulty);
```

- The this.blockchain.getDifficulty() returns the hex value of the current blockchain's index difficulty. This value is calculated by powering the initial difficulty by 5 every 5 blocks.
- The block.getDifficulty() returns the hex value of the first 14 bytes of block's hash and compares it to the currently accepted difficulty.
- When the hash generated reaches the desired difficulty level, it returns the block as it is.

B. Transaction Confirmation

The Miner gets the list of pending transactions and creates a new block containing the transactions. By configuration, every block has at most 2 transactions in it.

Assembling a new block:

- From the list of unconfirmed transaction selected candidate transactions that are not already in the blockchain or is not already selected;
- Get the first two transactions from the candidate list of transactions;
- Add a new transaction containing the fee value to the miner's address, 1 per transaction;
- Add a reward transaction containing 50 or more coins to the miner's address;
- Prove work for this block.
- Finally the transactions get added to chain as block.

VII. Network

MyBitcoin is a decentralized, open-source blockchain-based operating system with node server functionality, proof-of-stake principles as its consensus algorithm and a cryptocurrency native to the system, known as MyBitcoin

MyBitcoin is holding the protocol with its own blockchain. MyBitcoin declared its independence with the creation of the Genesis block in 2022.

MyBitcoin is using a total market cap of about $1 billion while launching the coin in 2022.

VIII. Incentive / miner reward

Being a lightweight cryptocurrency, mining (process of payment verification) is equitably distributed. A predetermined value of percentage of mining is set for individual miners and algorithms are in place prevent collusion. No mining operation can be backtracked i.e. once a transaction take place it can't be removed from blockchain or undone.

A reward Transaction will be created once the miner completes the mining process successfully and he will be rewarded in satoshis format.

Here we are using ramda, one that makes it easy to create functional pipelines.

A. Code Snippet:

```
//Importing ramda
const R = require('ramda');

// Creating a transaction list using ramda
const transactionList = R.pipe(
            R.countBy(R.prop('type')),
            R.toString,
            R.replace('{', ''),
            R.replace('}', ''),
            R.replace(/"/g, '')
          )(baseBlock.transactions);
```

IX. Reclaiming storage

Once a transaction goes through, the blockchain has a new addition. To prevent overburdening of storage, all the previous transactions are archived

and pushed to archives by using data compression techniques and retrieval systems.

X. Splitting of Value

A. Bits

Bits is the smallest unit of the cryptocurrency MyBitcoin. The Bits to MyBitcoin ratio is 100 million Bits to one MyBitcoin.

The Bits represents one hundred millionths of a MyBitcoin, and it helps to perform smaller transactions.

For instance, assuming MyBitcoin to USD is currently valued at 50000 USD and if you purchase a sports-shoe that has a current market value of 200 USD, you would have to shell out 400000 Bits or .004 MyBitcoins.

B. How Is Bits Different From Other Digital Denominations?

Many cryptocurrencies use denominations specific to their designer's preferences. For example, Bitcoin uses only the satoshi as a denomination, while Ethereum uses several.
Here we are using **BITS** as a denominations.

C. How Many Dollars Is 1 Bits?

Bits value changes with the market value of MyBitcoin. On Feb. 22, 2022, MyBitcoin had a market spot price of $50.55. At that time, one Bits was worth $.0000005055.

D. How Can I Buy Bits?

If you're only looking to exchange money for cryptocurrency, you can buy Bits on an online cryptocurrency exchange. Most exchanges list markets for several cryptocurrencies and their associated denominations, with an option to purchase or sell them

XI. Privacy / Encryption

- MyBitcoin uses Secure Hashing Algorithm (SHA) -256 algorithm. This algorithm generates verifiably random numbers in a way that requires a predictable amount of computer processing power.
- SHA 256 generates the **output will always be 256-bits in length**.
- The process of hashing is **not a method of encryption** as it is only a **one- way process** and therefore **cannot be reversed** (decrypted).
- By running multiple outputs through SHA-256, we can see how different the output becomes, even when only changing a single character in the message.
- We are using crypto module provides cryptographic functionality that includes a set of wrappers for OpenSSL's hash, HMAC, cipher, decipher, sign, and verify functions.

A. Code Snippet:

```
const crypto = require('crypto');
//Crypto utill class will generate a hash code based on sha
256 algorithm
class CryptoUtil {
      static hash(any) {
        let anyString = typeof (any) == 'object' ?
JSON.stringify(any) : any.toString();
        let anyHash = crypto.createHash('sha256')
        .update(anyString).digest('hex');
        return anyHash;
      }

      static randomId(size = 64) {
              return crypto.randomBytes(Math.floor(size / 2))
              .toString('hex');
      }
}
```

.

XII. Setup and Operation

HTTP Server the starting point to interact with the MyBitcoin, and every node provides a swagger API to make this interaction easier.
From the Swagger UI is possible to access a simple UI to visualize the blockchain and the unconfirmed transactions.

To Setup the Server in local machine,

Cloning repository

```
$ git clone git@github.com:Cryptocurrency-test/Testcoin.git
$ cd Testcoin
$ npm install
```

(Need to update the repository to MyBitcoin once after the confirmation.)

To Test the Functionality of the Server in local machine,

Testing

```
$ npm test
```

To Start the Server in local machine,

Run a node

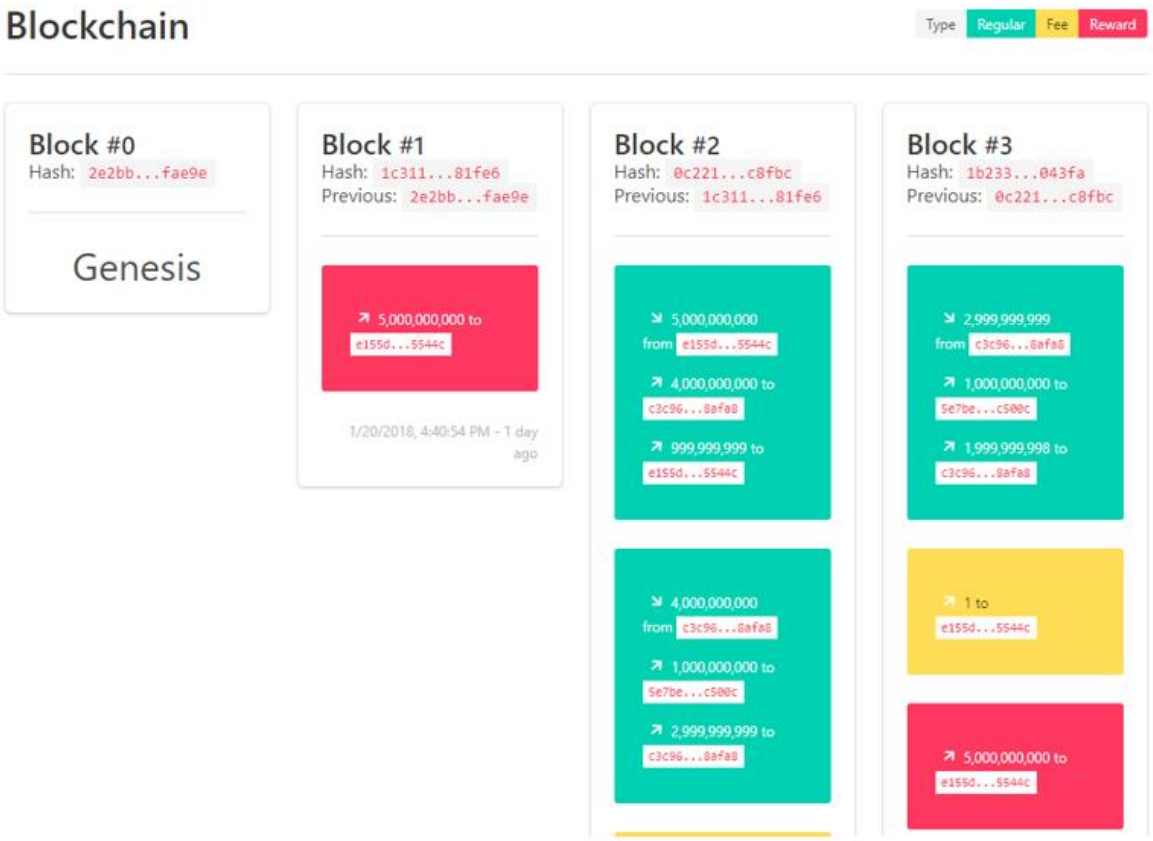
```
$ node bin/naivecoin.js
```

Run two nodes

```
$ node bin/naivecoin.js -p 3001 --name 1
$ node bin/naivecoin.js -p 3002 --name 2 --peers
http://localhost:3001
```

Access the swagger API

http://localhost:3001/api-docs/



Currently planning to launch the server with the help of AWS – EC2 Instance.