PROJECT PRESENTATION

BlockChain Project

By : Yossef Davarashvili & Jenya Zalkind

Implemented and missing components :

Implemented:

- Implemented MerkleTree.
- Implemented BloomFilter.
- Transaction validation.
- Main wallet + two node wallets.
- Mining reward.
- Each block - 4 transactions. (Slicing)
- func to count how much coins burned + Mined.
- Each Wallet start with minning reward of 100 coins.
- Burning 20 coins for each mine reward.

Missing:

- Segwit.

Known Bugs:

- Merkletree validation.

Environment setup:

Requirements for environment :

- npm install merkletreejs
- npm install bloom-filter
- npm install elliptic
- npm install cryptojs/SHA256
- npm install secp256k1
- npm install nodejs

Every Block contains 4 transactions :

```
console.log(this.block_transactions)
this.block_transactions.push(reward_tx)
this.block_transactions.push(reward_tx_for_burn)
const block = new Block(Date.now(),this.block_transactions,this.getLatestBlock().hash)
block.mine_block(this.difficulty)
block.initializeBloomFilter(this.block_transactions)
block.initializeMerkleTree(this.block_transactions)
console.log("##### Block successfuly minded ######")
this.chain.push(block)
// slicing the pending transactions to 4
this.pending_transactions=this.pending_transactions.slice(4,this.pending_transactions.length)
```

Ex of slicing blocks into 4 transactions:

MerkleTree & BloomFilter:

Functions inside Block class to initialize the wanted functionality :

```
initializeMerkleTree(transactions){
    //first lets map the transactions
    const leaves = Object.entries(transactions).map((x) => SHA256(x.signature))
    // now lets build the merkle tree
    this.merkletree = new MerkleTree(leaves,SHA256)
    this.root = this.merkletree.getRoot().toString('hex')
}

initializeBloomFilter(transactions){
    for (const tx of transactions){
        if (tx.fromAddress != null){
            this.BloomFilter.add(tx.signature)
        }
    }
}
```

Functions inside Block class to search and verify and validate :

```
foundInBF(signature){
    // is the signature can be found in the bloom filter
    //return Bool
    return this.BloomFilter.has(signature)
}

foundInMT(signature){
    // is the signature can be verified by merkle tree
    // return Bool
    const leaf = SHA256(signature)
    const proof = this.merkletree.getProof(leaf)
    return this.merkletree.verify(proof, leaf, this.root)
}
```

Verifying transaction:

Functions that validates the transaction :

```
has_valid_transaction(){
    for (const tx of this.transactions){
        if(!tx.is_valid()){
            return false
        }
    }
    return true
}
```

The log :

```
-------BlockChain Chain is valid ?-------
Valdating blocks
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Yes
```

Verifying transaction:

Also transactions can be validated in merkletree or bloom filter :

```
console.log("BloomFilter validation answer -> " + JCoin.getLatestBlock().foundInBF(JCoin.pending_transactions[22]['signature']))
console.log("MerkleTree validation answer -> " +JCoin.getLatestBlock().foundInMT(JCoin.pending_transactions[22]['signature']))
```

The bug mentioned before :

```
##########################################################################
BloomFilter validation answer -> true
MerkleTree validation answer -> false
```

Mining block and mining reward + burning coins:

```
class BlockChain {
    constructor(){
        this.chain=[this.createGenesisBlock()]
        this.difficulty=2
        this.pending_transactions=[]
        this.mining_reward=100
    }
```

Each miner get mining reward of 100 coins and from each mine the blockchain will burn 20 coins.

```
const reward_tx = new Transaction(null,mining_reward_addr,this.mining_reward - 20)
const reward_tx_for_burn = new Transaction(null,'burning_wallet', 20)
```

Burning coins :

Validating the amounts of burned + Mined + Total (Burned coins checked by 2 methods) :

```
Balace of 'burning_wallet' is 80
Num of coins that burned : 80
Num of coins that mined : 400
```

Total amounts in wallets:

```
------- Balaces of all wallets ------
Balace of 'MyWallet' (Main Wallet) is 155
---------------
Balace of 'addr2' (Node1 Wallet) is 75
---------------
Balace of 'addr3' (Node3 Wallet) is 90
---------------
Balace of all the coins in the blockchain 400
---------------
```

Transaction Log:

functiom that creates random 30 transactions fron the wallets :

```
keys_arr = [my_key,key_2,key_3]
wallet_addr_arr =[my_wallet_address,address_2,address_3]

for(let i=0;i<31;i++){
    rand_from = randomInt(0,2)
    rand_to = randomInt(0,2)
    rand_ammount = randomInt(1,5)
    const tx = new Transaction(wallet_addr_arr[rand_from], wallet_addr_arr[rand_to],rand_ammount)
    tx.sing_transaction(keys_arr[rand_from])
    JCoin.add_transaction(tx)
}
```

The log that shows that each transaction is valid :

```
Block mined - Num of Nonce needed -> 204
##### Block successfuly minded ######
[]
Block mined - Num of Nonce needed -> 101
##### Block successfuly minded ######
[]
Block mined - Num of Nonce needed -> 136
##### Block successfuly minded ######
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
Validating transaction
```

*Added additional .txt file that saved all the transactions.

Log :

```
[]
Block mined - Num of Nonce needed -> 204
##### Block successfuly minded ######
[]
Block mined - Num of Nonce needed -> 101
##### Block successfuly minded ######
[]
Block mined - Num of Nonce needed -> 136
##### Block successfuly minded ######
```

log that shows the mining and nonce needed for each block + message that the block mined successfully.

Example of transactions in log :


```
[
  Transaction {
    fromAddress: '04886c2982b9b80b081a5314c2169e214d1092f6ede56ffdcab08624b959f5485688e8a292eaf408a431386299f1b13a4969ac13fd42ce29c2b6e517f79540c318',
    toAddress: '04886c2982b9b80b081a5314c2169e214d1092f6ede56ffdcab08624b959f5485688e8a292eaf408a431386299f1b13a4969ac13fd42ce29c2b6e517f79540c318',
    amount: 4,
    time_stamp: 1672487496932,
    signature: '304502206685c375e8e7dad387eb1ddaeb9beff1879e438d8ea307e96a69583580dcc922022100eacb3d95a28415d6bab7a32068716cac367211f2dd21e2f75d3ece67eba93d80'
  },
  Transaction {
    fromAddress: '0486c585c7da1222e8a04017f553279e6ec09f959d9e1e21d2e71795e6cedee4e1b2a46ac1c4b49eefc53bcf33e6a2727e2bcb9d145f730263e25dc7693d9a6560',
    toAddress: '04886c2982b9b80b081a5314c2169e214d1092f6ede56ffdcab08624b959f5485688e8a292eaf408a431386299f1b13a4969ac13fd42ce29c2b6e517f79540c318',
    amount: 5,
    time_stamp: 1672487496939,
    signature: '3045022100d73b313b10921c9487898e257eb43c1e3eecac0ecb8e89ebda18b26158a3da4902203fc8993602791ebf8e03d028df2aae039d0f0753ae1fee385479cb2d10f9822d'
  },
  Transaction {
    fromAddress: '04d0e23ed0cf5ba2631d3845633bf6e0e6e4640179176e951bce4ac5e17ab81245f976261fc910e0811885f43b569459ce0b1c71f1f9bac94b92a8584b4187c81a',
    toAddress: '0486c585c7da1222e8a04017f553279e6ec09f959d9e1e21d2e71795e6cedee4e1b2a46ac1c4b49eefc53bcf33e6a2727e2bcb9d145f730263e25dc7693d9a6560',
    amount: 3,
    time_stamp: 1672487496943,
    signature: '304402203f3ecf907d4939208bd0a6bcc8ab251feed17f4f5e0cd1787b542422f2bf14bd02204fecdaf2c5025d437869e40d3f1405e5d1dda076c82755e3d8837dbcdaa11073'
  }
]
```