**CSE 598 – Engineering Blockchain Applications - Project2 Report**

**Title**: Dash Public Blockchain and cross-chain interoperability

**Student Name:** Tiriac Ioana-Raluca

**ASU ID:** 1217060520

**Abstract:**

The project focuses on creating a financial transaction using the Dash framework, known for its fast transaction confirmation time. A transaction has to be prepared that sends money on the Dash blockchain from one user wallet to another.

**Keywords:**

Dash, wallet, transaction, UTXO, satoshi, chainrider

**Introduction:**

Transaction speed is one of the main differences between Dash and Bitcoin. While it can take 20 minutes to get a transaction confirmation in Bitcoin blockchain, Dash offers near-instant transaction speed. The assignment for this project part allowed us to test this feature on Dash’s testnet, by preparing a transaction that sends a certain amount of Dash from one user wallet to another.

**Terminology:**

**Dash:**

Dash is an open source cryptocurrency, forked from the Bitcoin protocol, it’s a decentralized autonomous organization (DAO) and stands for digital cash. “Move Dash in a second for less than a cent, any amount, anywhere, anytime”[[1]](#footnote-1), Dash permits untraceable transactions.

**e-wallet:**

A digital wallet is a financial account, that stores information about the user’s funds, his private keys and payment history.

**transaction:**

A transaction on the blockchain is a financial exchange between two parties, a sender and a receiver and has to be validated before being recorded on the blockchain.

**UTXO:**

UTXO stands for Unspent Transaction Output, meaning it’s the change left over from another transaction/s, that has not been spent yet. The UTXO model allows for efficient parallel processing of transactions.

**Satoshi:**

The satoshi is the smallest unit the bitcoin unit = 10^-9 and was named after the creator of the protocol used in blockchains and the bitcoin cryptocurrency and author of the bitcoin white paper, Satoshi Nakamoto.

**Chainrider:**

Chainrider provides an ecosystem of tools and systems built around private and public blockchain that help with developing blockchain applications.

**Goal Description**:

**Phase I:**

1. Perform a get request to get the UTXO object for a user wallet, given the sender’s address and his token
2. Create a transaction using the dashcore library and serialize it
3. Send the transaction using Chainrider’s ‘*Send Raw Transaction API’*
4. The result, in form of transaction ID, was submitted to the auto grader

**Description of proposed solution:**

**General Information:** Libraries used in the package.json of the project were : dashcore-lib, got and request. Requests were made from the browser and using “got” for the GET request, while the post request was made using the “request” npm package.

**Phase I:**

1. The Unspent Transaction Output (UTXO) is simply the user’s change left over from other transactions created in form of an output. In order to get a user’s (who has an address) UTXO object on chainrider a get request must be made using the URL:

let url = `https://api.chainrider.io/v1/dash/testnet/addr/${sender}/utxo?token=${token}`

The token is simply an API token, every registered user of the Chainrider API gets to have one to be able to issue API web requests and the sender here signifies the user’s (future coin sender) address.

The GET request can be made from the browser or using any js library that provides capabilities to issue web requests, like this :

(async () => {  
 try {  
 // GET request  
 const response = await got(url);  
 ***console***.log('GET response:', response);  
 } catch (error) {  
 }  
})();

The response is not synchronous so it has be executed inside an async() and be awaited. The UTXO object in JSON format looks like this :

let utxo\_obj = {"address":"yTYZjnTuepHbVAcoWq4g7f5teXru4KSJMa",  
 "txid":"e9b25b28e08c86a3d1658b1bd5138356f502a863f25f1ac2d73f6440192a62b0",  
 "vout":1,  
 "scriptPubKey":"76a9144f4409f6a42a8f3ff565e4d1df47e8e8b1d509cb88ac",  
 "amount":0.00039,"satoshis":39000,  
 "height":401803,"confirmations":31};

The UTXO object contains an address, the transaction id (txid) that issued this output (so the transaction that had this output as “change”), the public key of the UTXO object(the lock) and the amount in satoshis. Satoshi is the smallest unit of a bitcoin, it is 0.00000001 BTC. To compare cryptocurrencies : while 1 BTC has 100,000,000 satoshis, 1 Dash has 473,075 satoshis.

|  |
| --- |
|  |

1. Next, a transaction was created using the dashcore library, to send a given amount from one user’s wallet to another. This transaction received as input the UTXO object obtained previously which has the financial unspent funds, the receiver’s address, the amount to send and the address to send the change of the transaction to.

let transaction = new dashcore.Transaction()  
.from(utxo\_obj)  
.to(receiver, 20000)  
.change(sender)  
.sign(senderPrivatekey)  
  
rawtxserial = transaction.serialize();

In order for the transaction to be verified it has to be signed with the private key of the sender, which is the key to the “lock” (the public key) of the UTXO object. Only the owner of the UTXO output has the key to its lock.

The chainrider API requires submitting a raw transaction, so it had to be serialized before posting it.

1. The serialized transaction was sent to the blockchain’s testnet by forming a POST web request using the ‘request’ npm package.
2. const options = {  
    url: 'https://api.chainrider.io/v1/dash/testnet/tx/send',  
    json: true,  
    body: {  
    rawtx: rawtxserial,  
    token: token  
    }  
   };  
   request.post(options, (err, res, body) => {  
    if (err) {  
    return ***console***.log(err);  
    }  
    ***console***.log("the body is")  
    ***console***.log(body);  
   });

**The options part** mentions the URL = dash’s testnet, the body to be sent – the serialized raw transaction, the token for the chainrider API and the format of the content to be sent -JSON format. The response to this request is the transaction ID, also meaning the transaction was submitted successfully to the blockchain.

**Issues Faced and Methods used to resolve them:**

At first made requests did not work because error handling and asynchronous responses were not taken into account. After some research the correct way to handle web requests in javascript code was found.

**Conclusions**:

This project complemented my understanding about user wallets and transactions on the blockchain. It also awakened my interest about other cryptocurrencies other than Bitcoin and definitely taught me how to write correct web requests using js libraries.

**Bibliography**:

<https://www.genesis-mining.com/dash-bitcoin-difference>

<https://www.chainrider.io/docs/dash/>

<https://www.npmjs.com/package/got>

<https://www.dash.org/>

1. www.dash.org [↑](#footnote-ref-1)