CSE598 Engineering Blockchain Applications

Project 2: Dash public blockchain and cross-chain interoperability.

This document contains description of tasks required to complete PROJECT 2 assignments. This project will help you familiarize with the Dash public blockchain ecosystem by helping you understand, write and execute cryptocurrency-based financial transactions.

For this project you will be working with Dash public blockchain and Hyperledger Fabric frameworks. Dash is a P2P payments network and cryptocurrency that is used to record transactions on a blockchain. In this project you will use a Dash library in NodeJS programming language for writing transactions on the Dash blockchain network (the testnet). Read more about Dash blockchain network on this link (https://dashplatform.readme.io/docs/introduction-what-is-dash)

Financial transactions on public blockchains

Bitcoin is the most popular public blockchain out there for sending payments. However, while Bitcoin might be the oldest option, it is most certainly not the best because of its transaction confirmation time. Dash, on the other hand thrives in fast transaction confirmation and is our choice for this project because of it's applicability in a broad range of industrial use-cases concerning fast digital payments.

Transactions enable users to send and receive payments: on the Dash blockchain the cryptocurrency is named Dash. A transaction has at least one input and one output. Each input spends the dash amount paid to previous output and each output then wait an Unspent Transaction Output (UTXO) until a later input spends it. Read more about the transactions on this link (https://dashcore.readme.io/docs/core-guide-transactions). Dash core transaction library at this link https://dashcore.readme.io/docs/core-ref-transactions provides detailed description of attributes of the transaction.

Wallets: Blockchain wallets store and manage keys used to interact with the blockchain. Because transactions need a digital signature to spend an output, which requires knowledge of a private key, blockchain wallets are one of the most important applications used with a blockchain. The secure storage and management of keys is crucial for most transactions.

In this project, you will learn to write two types of transactions on the Dash blockchain network:

- Financial transactions that send money on the Dash blockchain from one user (wallet) to another.
- Data transactions that write certain information on the Dash blockchain (publicly accessible).

This project is divided into two parts: In part 1, you will need to prepare one transaction that sends a certain amount of Dash cryptocurrency from sender to receiver wallet.

In part 2, you be using the smart contract from Project 1 to establish cross-chain communication [Hyperledger Fabric to Dash and vice-versa]. This part of the project is optional. For writing transactions of the project, you will receive a code base that needs to be updated on certain places so that the transaction is complete and recorded on the dash blockchain network.

Project 2: Part 1-Writing a financial transaction to send some Dash between two wallets

TASK 1: Write a function to send some Dash from sender to receiver wallets

In the provided code base, sendMoney.js file contains the address of the sender and authentication token. Get the amount available with the sender by accessing the URL in the sendMoney.js. If the amount available

with the sender is greater than or equal to the send amount given in the sendMoney.js create a transaction using dash core library and send the transaction using chainRider send raw transaction API (https://www.chainrider.io/docs/dash/#send-raw-transaction). Get the resulting transaction ID and submit it to the grading server.

Submission:

Submit the transaction ID.

Project 2: Part 2 Hyperledger Fabric to Dash interoperability

The goal of this project is to advance your knowledge of how two independent, completely different blockchain frameworks might collaborate and why. Project 1 dealt with patient records, and it was inside a private blockchain. That means no outsider without proper access can see the information. Imagine this patient records system being installed in a lot of hospitals that want to keep most of the data private, however, some information should be public for other hospitals/similar institutions to see. The system designer suggests that patient records will be written to the private blockchain (Hyperledger Fabric) in every hospital, however, if a patient is registered with a rare blood type (in our case AB-: the rarest blood type in the world), then this information is going to be written on the Dash blockchain (testnet) public for all other health institutions to see. This might help speed up the search for rare blood type donors in case of emergencies.

In the code base, precord.js file contains getter and setter methods for patient record. It also contains createInstance() method to create a new Precord. The file irecordlist.js contains the code to add, get and update Precord. Precordcontract.js file contains the smart contract for the patient record (Precord). In this project you will be completing the functions of Precord class and PrecordContract class.

TASK 1: Rewrite the createPRecord function (precordcontract.js)

Instructions:

Update the createPRecord function to write the OP return transaction to Dash public blockchain network. This function takes transaction context, username, name, dateofbirth, blood_type, base_url, address, private_key and token as the input. Create an PRecord with username, name, dob, gender, blood_type and transaction ID(none). If the patient bloodType is **AB-** insert record to the blockchain using addPRecord() function of PRecordList. Fetch the transactions for the address{address:base_url, address, token}. Compute the total amount of all the raw transactions fetched. If the total amount is greater than or equal to amount+fess create a transaction using the dashcore library and send the transaction using ChainRider. Learn more about send raw transaction API on this link

https://www.chainrider.io/docs/dash/#send-raw-transaction. Use the resulting transaction ID (dashTx) to create PRecord.

Task 2:

Complete the **queryWithQueryString()** function of the PRecordContract class. When iterating through records check if the record has a TX field. If YES, get OP-Return data from the transaction. Convert the hex value of OP return data to ASCII and append it to the record as jsonRes.Record[‘tx_decoded’].