DataSys Coin (DSC) Blockchain

# Implementation:

Classes:  
1. Wallet:

This class is a basic wallet implementation that interacts with a blockchain. It uses Flask framework (python) to create a web service through which user can access wallet, this class has following functionalities.

**Functionalities:**

#### Create Wallet:

To create a wallet, we have a function called create\_wallet.it creates a new wallet by generating key, value pair. It works as follow:

1. Checks if there is an existing wallet by searching for key files.
2. if wallet exists, it returns. Otherwise, it generates a new RSA key pairs i.e., public and private keys.
3. Then it hashes both the keys using Blake3 and saves both of the keys.

#### Display Keys:

This function displays the existing private, public keys of wallet.

#### Check Balance:

This function retrieves the wallet balance by contacting the blockchain server. It works as follow:

* + 1. Read public key from the key file.
    2. Contacts the Server to get the wallet balance.
    3. Displays the wallet balance.

#### Send Transaction:

This function sends an amount/transaction to a recipient. It takes the user input for the amount to send, and also the address where it should send. It generates a new transaction ID. Then it sends the transaction id to the pool. Then it waits for the confirmation from the pool and changes the status to confirmed.

#### List Transaction:

This function displays the list of transaction. This function contacts the pool server to get the list of transactions, and then display that list.

2. Validator:

This class is an implementation of the validator in blockchain, that performs validation tasks based on the specified proof algorithm. Threading is done in this class to allow concurrent execution of the validations. It uses Metronome and blockchain server for fetching the necessary data. Following are the main functionalities of this class:

**Functionalities:**

#### Initialization:

Validator is initialized with a public key, proof algorithm to be used, blockchain server information, Metronome Server information and the fingerprint. It also configures the path for proof of space algo.

#### Validation Loop:

This function runs continuously in order to perform block validation based on the proof algorithm. It runs on a separate thread, to perform validation continuously. It fetches the latest block’s hash and difficulty from the metronome server and performs the proof algorithm based on the configuration given in initialization. After every validation process it prints the information about it.

#### Proof Of Work:

This function implements the proof of work algorithm, it involves finding a nonce, which when combined with the block hash, difficulty level and the public key, produces a hash and checks whether the hash meets the required difficulty level of the network, and then print the relevant information.

#### Proof Of Memory:

This function implements the proof of memory algorithm. It focuses on memory intensive operations. This function generates and organizes large number of hashes in the memory, then it looks up for the nonce that satisfies the condition, if the hashes are sorted, function reports success otherwise failure of proof, then it prints the relevant statistics.

#### Proof Of Space:

This function implements the proof of space algorithm. It generates, organizes and sorts large number of hashes based on the provided configuration. Then method performs lookup to find nonce, it demonstrates the effective utilization of the required storage space. In the end function reports the success or failure of the proof.

3. Other components:

Other components up to this point such as Pool, Blockchain, Metronome are incomplete and serve only some randomly generated values to see how those components would work together. Currently, we have created only simple versions of them and will continue working on them to implement actual functionality for them and develop fully functional system. Below we are including some screenshots of outputs that we get when running some components together.

* Wallet generation and balance request.

A computer code with numbers and letters

Description automatically generated

* Response from the Blockchain component to the request of balance.

A screenshot of a computer error

Description automatically generated

* Sending money to random address as a test

### A computer code with numbers and letters Description automatically generated with medium confidence

* Transaction being acknowledged by the pool

A screen shot of a computer

Description automatically generated

* Metronome trying to submit a block but failing because the function in the Blockchain component is not developed yet.

A white background with black text

Description automatically generated