InternalContract for XDC price feed integration

InternalContract is the one which performs all the functions pertaining to blockchain connection, registering the feed's job, initiating the jobs to fetch individual feed results. We have abstracted all those complexities under the hood, which is our internal contract and created a Consumer contract to access the same.

End users should be able to use the "Consumer contract" and access the oracle accordingly.

Internal contract gets data from 4 data sources such as Cryptocompare, Coinmarketcap, Kucoin and Bittrue for XDC/USDT feeds and aggregates the price to return a final value.

We got an additional feature in this internalContract, which is getting the aggregate value for each request by passing the requestID. By this the end user will receive the value for their contract initiation, so we can avoid showing the old fetched value by other users.

consumerContract for XDC price feed integration

Please note, as a user you should have enough "#PLI" token to pay oracle fee in order to access the price feed. For instance, the provided consumer contract needs to be deposited with minimum 1 PLI to max 100. For which, we have provided a small widget right below.

The consumerContract needs to be "Approved" and "Deposited" before using it in actual. We created a widget for the "Approval" & "Deposit", which can be downloaded from github and the steps to install the widget with github link is given below.

Steps to use the widget

Step 1:

Clone the git package to your local pc

Git clone `https://github.com/GoPlugin/plugin-feeds-internal-contract.git`

Step 2:

On command line go to **plugin-feeds-internal-contract/client** folder and do

```
npm install
```

Step 3:

On command line and run below command to install dependencies

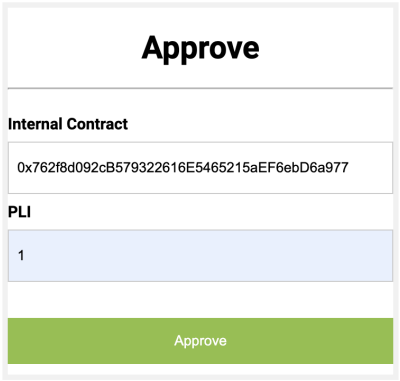
```
npm install
```

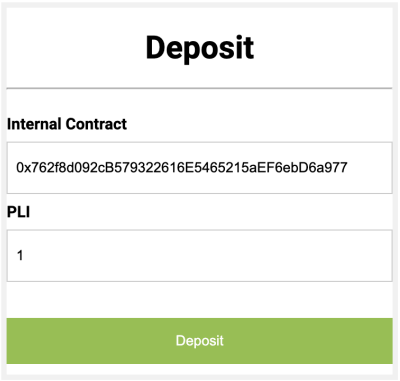
Step 4:

On command line initiate the widget using the below mentioned command

```
npm start
```

Once the widget is installed, the server interface opens up through http://localhost:3000. The UI is given below for the understanding.





Step 5:

In the widget Approve the “InternalContract” address **0x9bd7f00285956819A9f0d58C523585c6285742D3** , and then Deposit the “Internal Contract” using your Apothem PLI tokens from your wallet.

NOTE:

1) The minimum PLI token you can approve and deposit is 1
2) The maximum PLI token you can approve and deposit is 100

Based on the estimated usage you can deposit the PLI in between minimum and maximum, this PLI will be reserved in the InternalContract and whenever the end user use this feed, the ORACLE FEE of 0.1 will be deducted, in aggregation 0.1 * 4 source = 0.4 PLI will be deducted.

The consumer contract is given below for usage or you can get that in the repository under consumer-contract folder

```
pragma solidity ^0.4.24;

interface IInvokeOracle{
   function requestData(address _caller) external returns(uint256 requestId);
   function showPrice(uint256 _reqid) external view returns(uint256);
}

contract consumerContract{
   address CONTRACTADDR = 0x9bd7f00285956819A9f0d58C523585c6285742D3;
   uint256 public requestId;

   //Fund this contract with sufficient PLI, before you trigger below function.
   //Note, below function will not trigger if you do not put PLI in above contract
address
   function getPriceInfo() external returns(uint256){
       (requestId) = IInvokeOracle(CONTRACTADDR).requestData({_caller:msg.sender});
       return requestId;
   }
   //TODO - you can customize below function as you want, but below function will give
you the pricing value
   //This function will give you last stored value in the contract
   function show(uint256 _id) external view returns(uint256){
       return IInvokeOracle(CONTRACTADDR).showPrice({_reqid:_id});
   }
}
```

Steps to be performed post deployment:

Step 1:

When you deploy consumerContract.sol, you will get the external function “getPriceInfo” as mentioned below. Click on the “getPriceInfo” to initiate the price feeds, this action will perform a transaction using your XDCPay wallet. Once the transaction is done successfully, just wait for 30 seconds, so that the backend aggregation mechanism fetches the price and does the aggregation and writes it onto the blockchain.

Step 2:

Now, click on the “requestId” to get the request id for your specific feed request.

Step 3:

Copy the request id and paste it in the show field and click on “show” to get the aggregated XDC/USDT value.

Reference image of customerContract deployment