InternalContract for Proof of Reserve integration(Mainnet)

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.

The internalContract 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.

consumerContract for Proof of Reserve integration(Mainnet)

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.

Steps to use the widget

Step 1:

Clone the git package to your local pc

git clone **https://github.com/GoPlugin/POR-plugin-integration-mainnet.git**

Step 2:

On command line go to **POR-plugin-integration-mainnet/client** folder and do

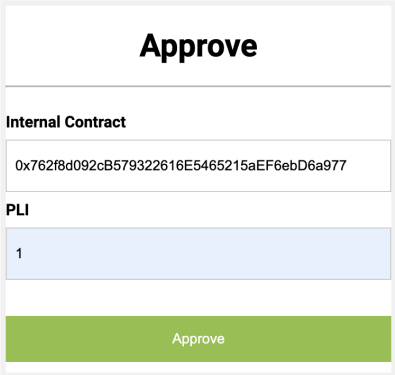
```
npm install
```

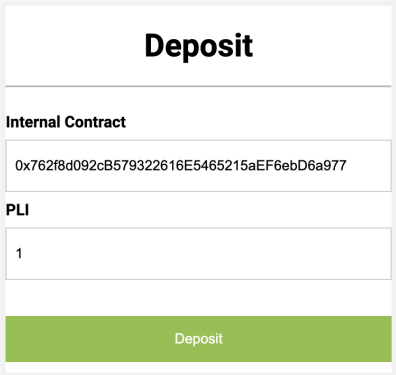
Step 3:

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.





NOTE: Please use the **same wallet address** of your XDCPay for all your transactions.

Step 5:

In the widget Approve the “InternalContract” address **0xa78FE51D7F9ca4EE4CfC38aEA3f0853F0E5C7AeD** , and then Deposit the “Internal Contract” using your PLI tokens from your wallet.

NOTE:

1) In the widget the minimum PLI token you can approve and deposit is 1
2) In the widget 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.

The consumer contract is given below for usage or you can get that in the repository under the consumer-contract folder. Make sure “CONTRACTADDR” variable matches with **‘0xa78FE51D7F9ca4EE4CfC38aEA3f0853F0E5C7AeD’,** which is the address of InternalContract deployed for Proof of Reserve.

Steps to be performed post deployment:

Step 1:

Deploy ConsumerContract.sol using remix.

NOTE: Please use the **same wallet address** of your XDCPay which you used to approve, deposit PLI in the widget.

```
pragma solidity ^0.4.24;

interface IInvokeOracle {
   function requestData(address _authorizedWalletAddress) external returns (uint256
requestId);
   function showPrice(uint256 _reqid) external view returns (uint256 answer, uint256
updatedOn);
}
contract ConsumerContract {
   address CONTRACTADDR = 0xa78FE51D7F9ca4EE4CfC38aEA3f0853F0E5C7AeD;
   uint256 public requestId;
   address private owner;
   constructor() public{
       owner = msg.sender;
   }
   //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) {
       require(msg.sender==owner,"Only owner can trigger this");
       (requestId) =
IInvokeOracle(CONTRACTADDR).requestData({_authorizedWalletAddress:owner});
       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, uint256) {
       (uint256 answer, uint256 updatedOn) =
IInvokeOracle(CONTRACTADDR).showPrice({_reqid: _id});
       return (answer,updatedOn);
   }
}
```

Step 2:

Once ConsumerContract.sol is deployed, collect the deployed address. The wallet address with which you deployed the ConsumerContract should be sent to the Plugin team.

Kindly send mail in below mentioned format to '**support@goplugin.co**'.

Mail Subject: Authorization requested for Proof of Reserve feed
Mail Content:
ConsumerContract:<your_deployed_consumercontract_address>
InternalContract address used: <internal_contract_address>
Wallet address:<your_wallet_address>

Plugin Team will authorize your ConsumerContract address, your wallet address and reply back to your mail request.

Step 3:

Once you receive a successful reply from the Plugin Team, go to your ConsumerContract.sol deployment and click on "**getPriceInfo"** function.

Step 4:

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 proof of Reserve value.

Reference image of customerContract deployment