Connect2Evolve

Token Engineering Challenge

MVP WORKSHOP

Diffusion Hackathon
Berlin 2019

SIEMENS

01

Challenges/Problems

A **Transparent system** for all stakeholders

B **Maximize usage** of the container capacity

C Transfer the **fractional ownership of the container**

D **Incentivize education and training** that leads to the global development

02

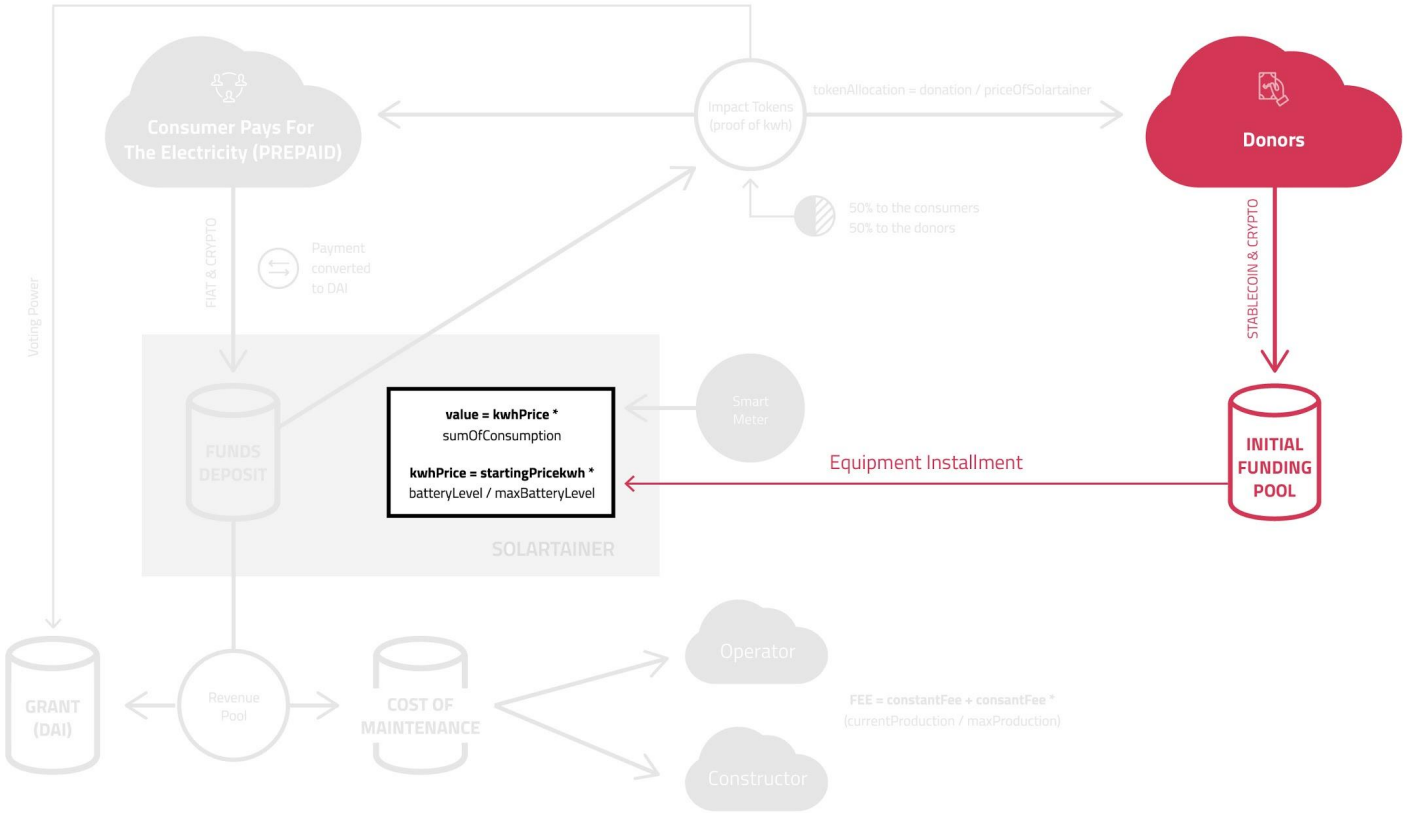
Solution

A **Token Model**
Here’s our token modeling canvas

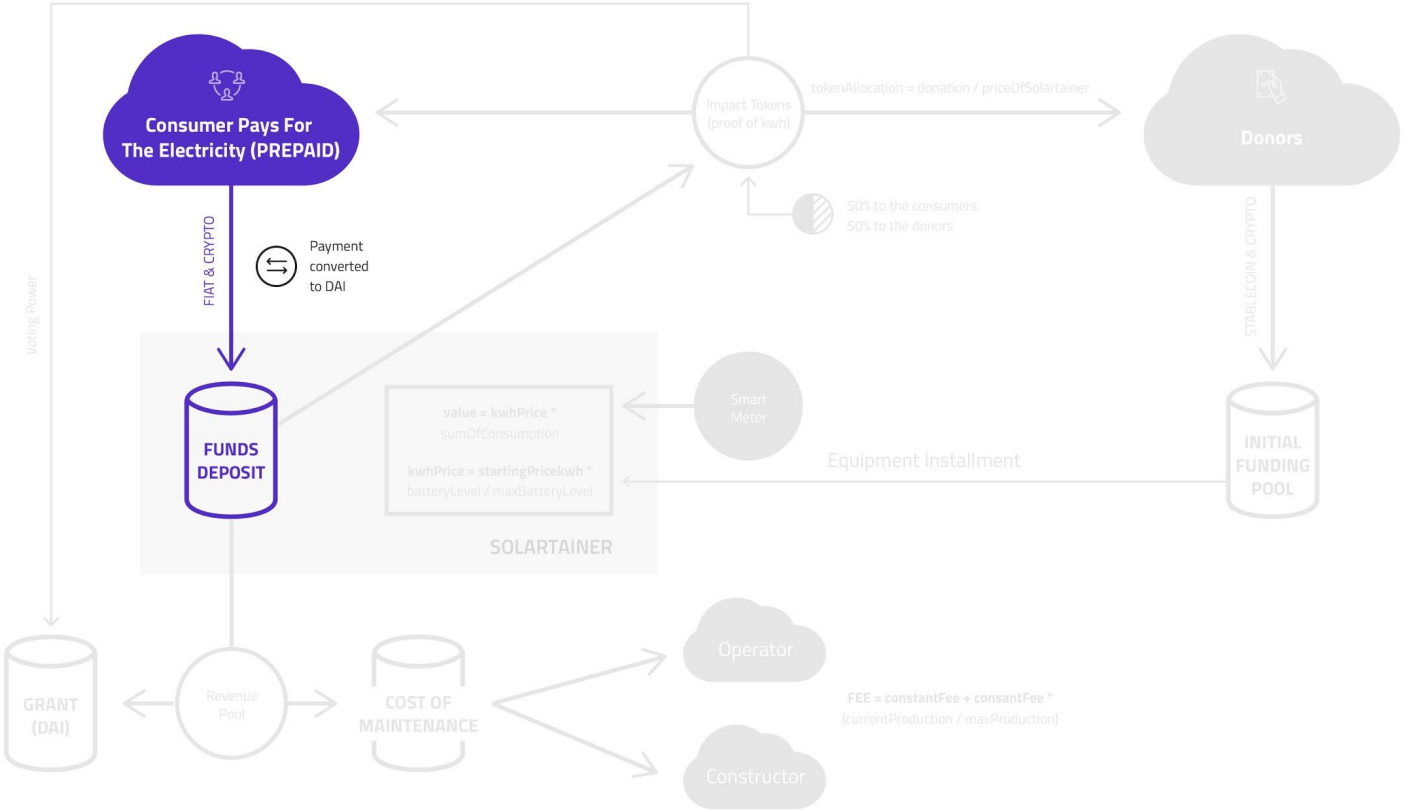
B **CadCAD simulation**
Available at GitHub

C **Smart Contracts**
Our GitHub repository

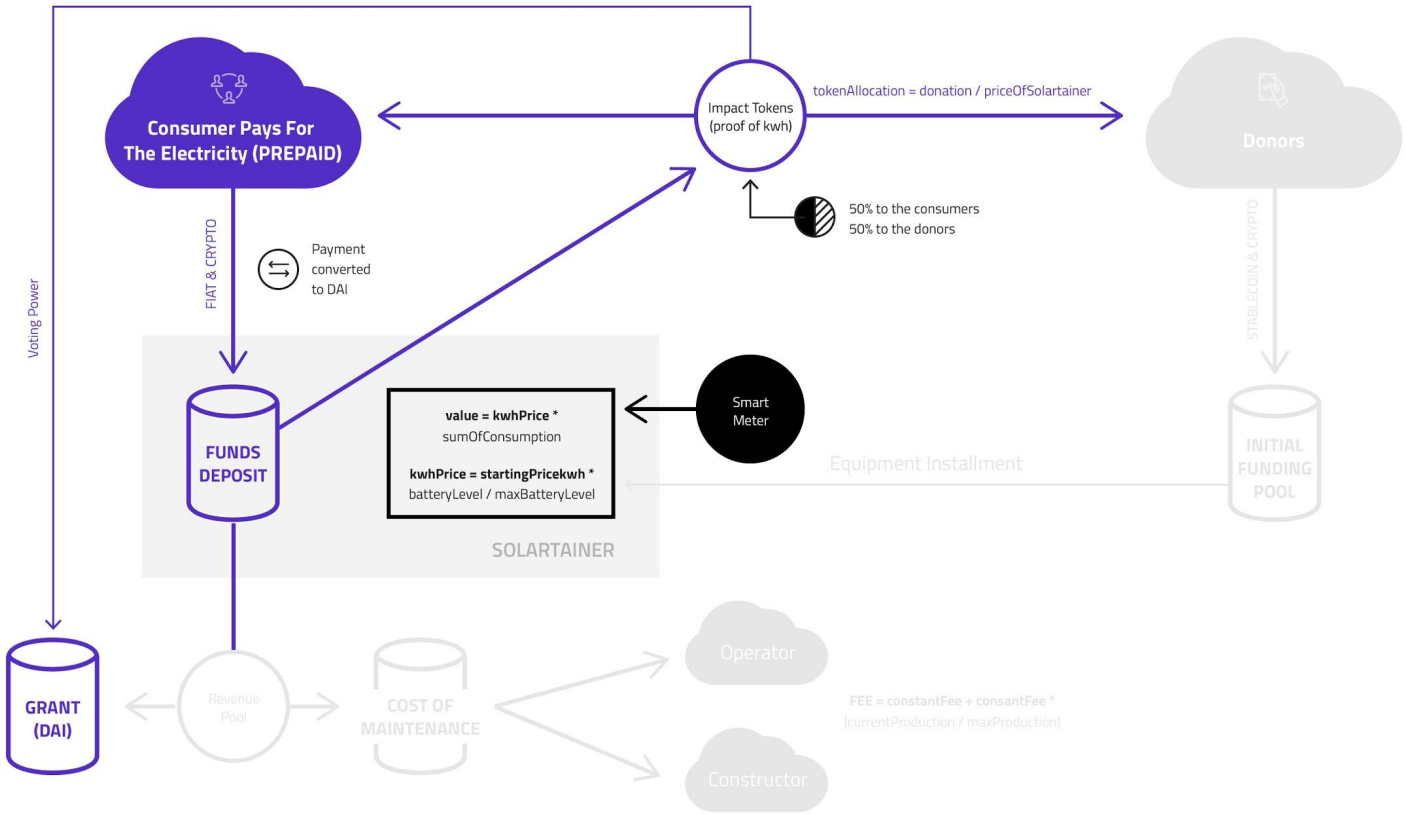
D **UI Dashboard**
diffusion.mvpworkshop.co



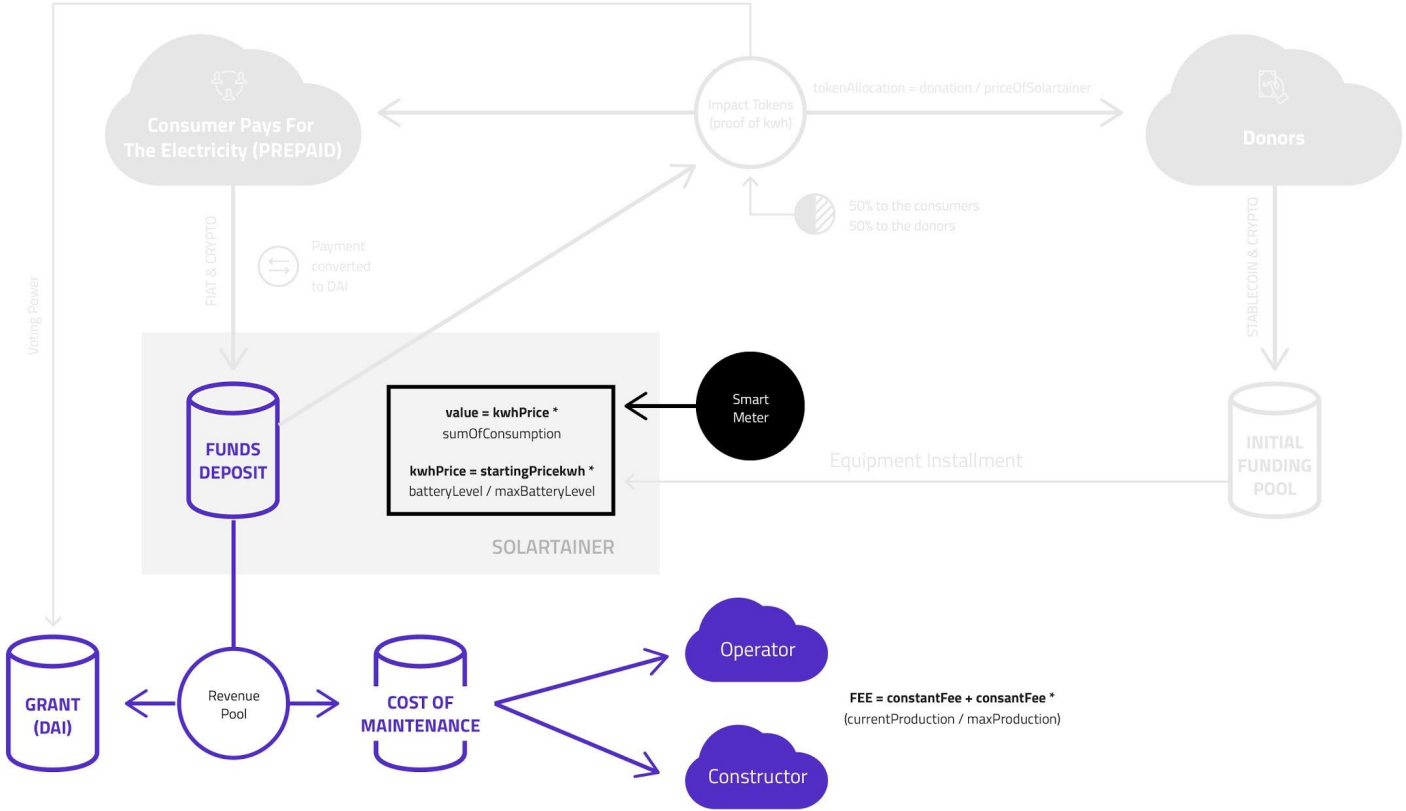
Token Model (1/5)



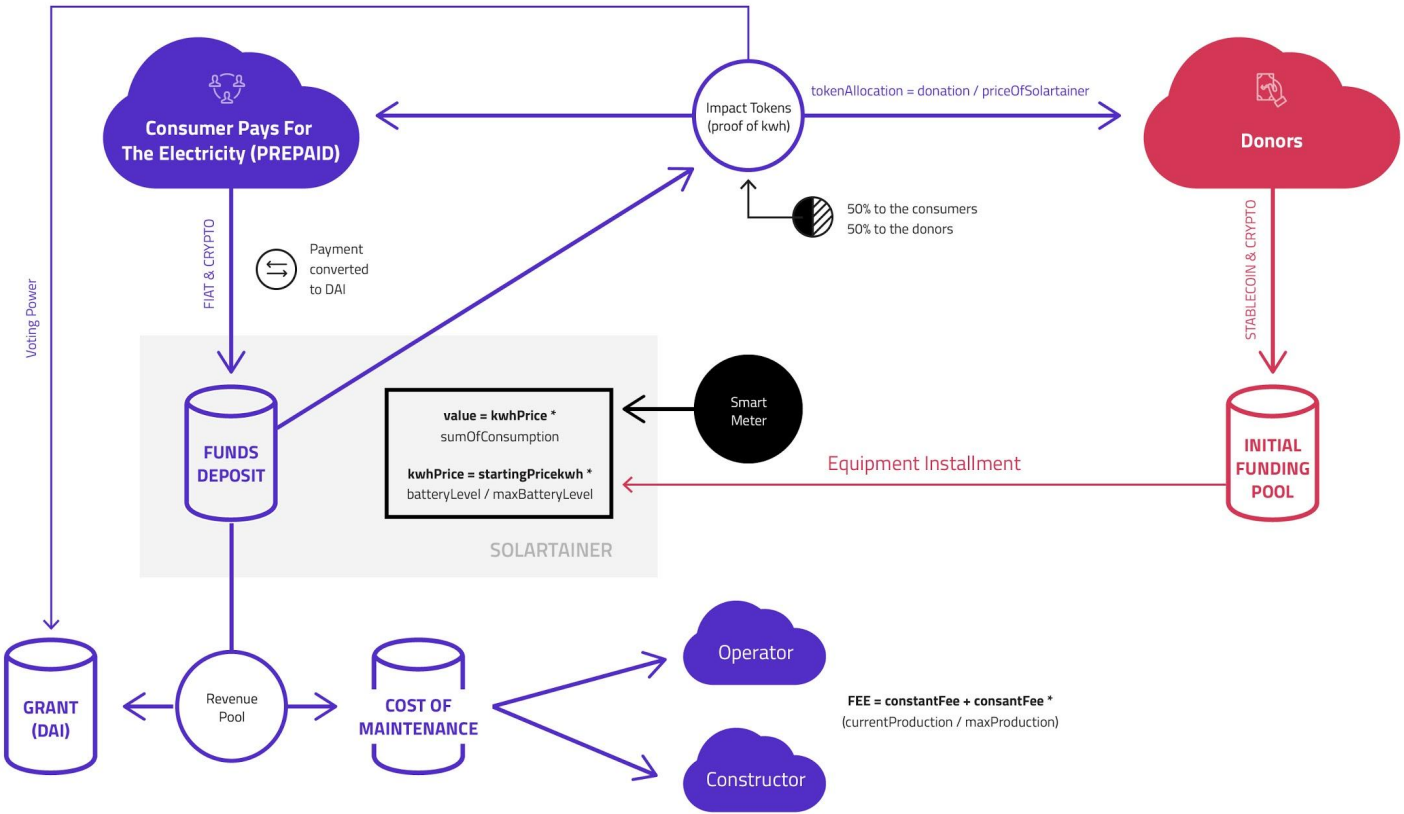
Token Model (2/5)



Token Model (3/5)

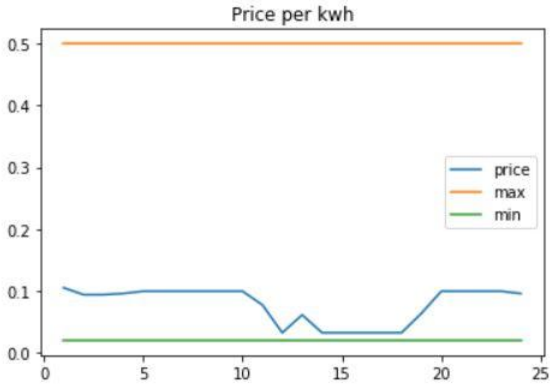


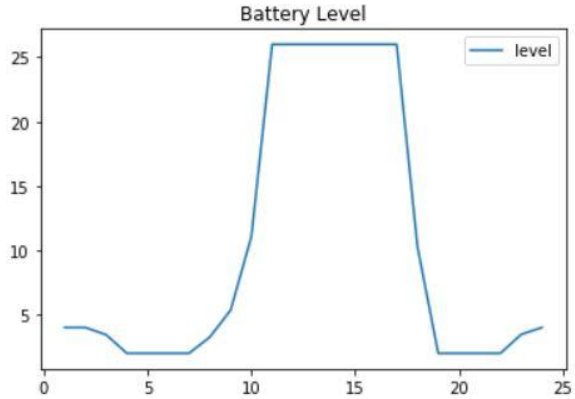
Token Model (4/5)



Token Model (5/5)

./blob/master/cadCADsimulation.ipynb





Variables

- ➔ `price_per_kwh`
- ➔ `max_price_per_kwh`
- ➔ `production_decay_rate`
- ➔ `maintenance_rate`
- ➔ `op_cost_per_hour`
- ➔ `op_cost_per_kwh`
- ➔ `maintenance_cost_per_time`
- ➔ **... more to be added**
  (such as weather forecasts)

CadCAD Simulation

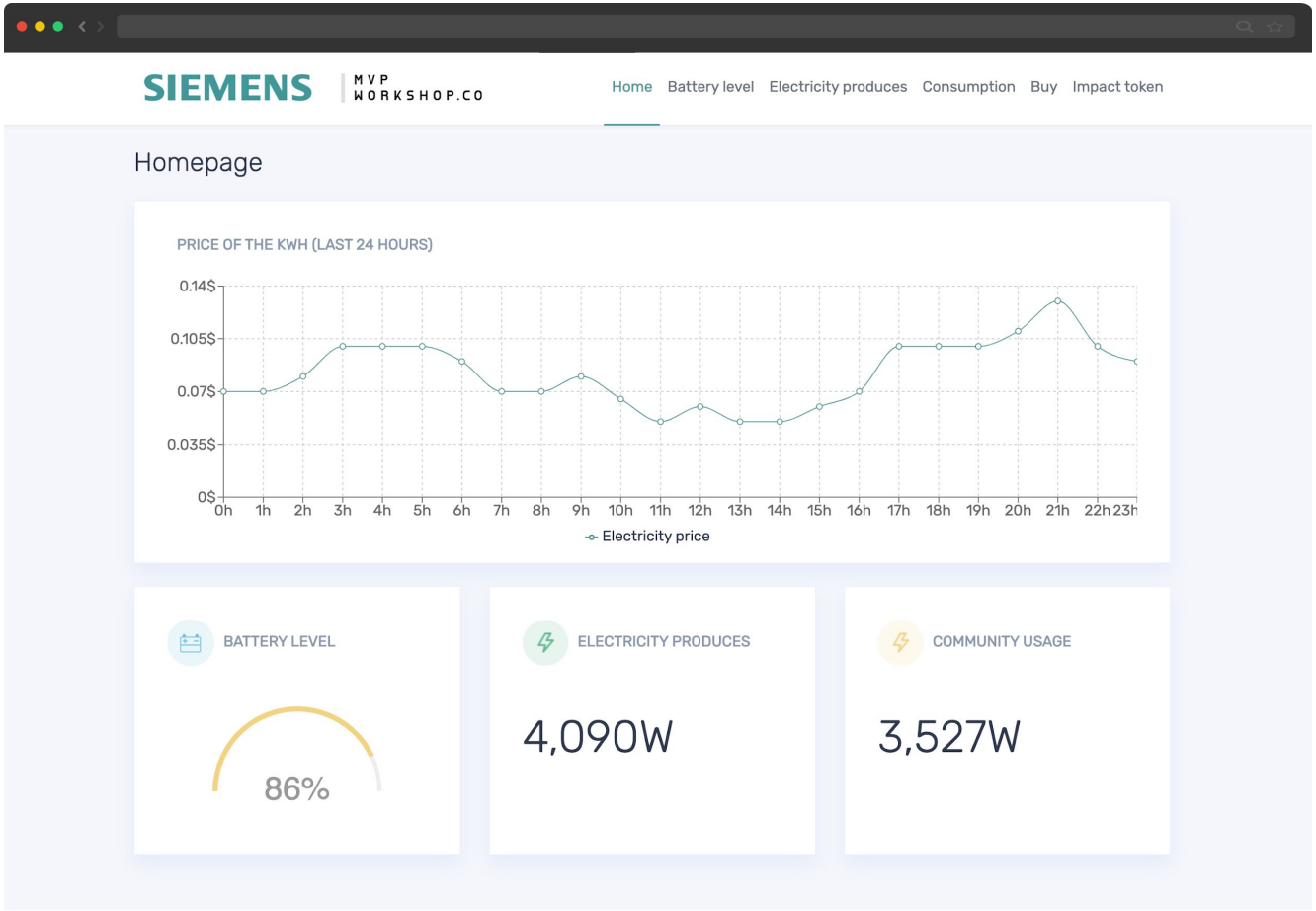
./tree/master/contracts

C

Smart Contracts

```
pragma solidity ^0.5.10;

import "openzeppelin-solidity/contracts/ownership/Ownable.sol";
import "./token/ImpactToken.sol";

contract Solartainer is Ownable {
    uint256 totalFundsDAI;
    mapping (address => bool) authorizedTokens;
    address BCY;
    mapping (address => uint256) communityBalancesBCY;
    uint256 currentTotalKwHour;
    uint256 maxTotalKwHour;
    uint256 startingPrice;
    uint256 maxPrice;
    address oracle;
    address impactToken;

    constructor (uint256 _startingPrice, uint256 _maxPrice, uint256 _maxTotalKwHour, address _oracle, address _impactToken) public {
        startingPrice = _startingPrice;
        maxPrice = _maxPrice;
        maxTotalKwHour = _maxTotalKwHour;
        oracle = _oracle;
        impactToken = _impactToken;
    }

    function authorizeToken(address token) external returns (bool) {
        authorizedTokens[token] = true;
        return true;
    }

    function prepayFunds(address token, uint256 value) external returns (bool) {
        // value is is `token` and should be converted to BCY with oracle
        require(authorizedTokens[token] == true);

        uint256 balanceBCY = value * 1; // rate should be variable

        communityBalancesBCY[msg.sender] += balanceBCY;
```

Consumers Dashboard



03

Next steps

- → **Presale - Fundraising Platform**
- → **Government Relations**
- → **Third Party API integration** (expand token simulation models, e.g. weather forecast)
- → **Infrastructure & Equipment Installation**

That's all

Thanks for having us at

Diffusion
2019

MVP·WORKSHOP