002 Partnership (PTR) Contract

Target release	
Epic	
Document status	DRAFT
Document owner	Alex Zabolotniy
Designer	Alex Zabolotniy
Developers	Alex Zabolotniy
QA	

General Economy Goals

- The goal of Solidity smart contract(s) is to provide the smart economy to Sample Protocol. more specifically to provide the below features:
  - User and data ownership registration, user permission management (Entity Management - EM Contract).
  - Data trading and client protection (Data Block Trading - DBK Contract).
  - Partnership organization, shares allocation for each data block and funds spreading (Partnership - PTR Contract).

Partnership (PTR) Contract Goals

- Manage shareholders and their shares:
  - Allocate shares at deployment stage
  - Transfer shares after allocation
  - Allocate more shares (dilute existing shareholders)
- Spread funds between shareholders of the contract (related to their shares).
- Calculate group reputation based on individual shareholders' reputation.

Background and strategic fit

Assumptions

-

Requirements

#	Title (type)	User Story	Importance	Notes

1	**Data types and variables**			

2	**Shareholder**
	(Data Type)
		Shareholder data structure represents a single owner in the contract.
		Shareholder.account *(address) - account number of the specific owner.*
		Shareholder.shares*(uint256) - allocated shares for the specific owner.*
			Must	Type: Data structure
				struct Shareholder{
				address account;
				uint256 shares;
				}

3	**shareholders** **(Variable)**	Stores an iterate-able list of shareholder accounts and their shares.	Must	*Type: Shareholder[ ]* *Attributes: private*
4	**holderAddressToIdx** **(Variable)**	Stores a mapping of shareholder address to its index in **shareholders** list.	Must	*Type: mapping (address=>uint256)* Attributes: private
5	**allocationRequestTable** **(Variable)**	Stores new shares allocation requests and given approvals per request.	Must	*Type:* mapping (addre => Shareholder[]) Attributes: private
6	**totalShares** **(Variable)**	Stores total amount of allocated shares	Must	*Type: uint256* *Attributes: private*
7	**EM**			
8	**Event**			
9	**LogDeposit()**	Log an event for depositing to the contract. Depositing means to transfer funds to the contract without calling any specific function.	Must	Arguments: (address _src,uint256 _value)
10	**LogChangeOwnership()**	Log an event when data-block ownership is updated	Must	Arguments: (address _src,address _dest,uint256 _value)
11	**Modifiers**			
12	**~~partnerOnly~~()**		**disabled**	~~Arguments: address _~~
13				
14				

15	**Functions**			
16	**constructor()**	Contract constructor what does it construct?	Must	Arguments: Sharehold _owners Modifiers: public, payable Returns: Disabled
17	**kill()**	~~consensus is required. say 80% must agree on the operation~~	**disabled**	
18	**()**	Payback function will call spreadFunds() function	Must	
19	**spreadFunds()**	Spreads total contract balance between shareholders. Each shareholder receives a proportionate share.	Must	Arguments: address _dest, uint256 _value Modifiers: public, payable Returns: bool (succes
20	**transferShares()**	Transfer shares. This function transfers already allocated shares and can't allocate new share. The sender is the function caller.	Must	Arguments: address _dest, uint256 _value Modifiers: public ,payable Returns: bool (succes
21	**addNewShareholder()**	Adds new shareholder. This function will dilute other shareholders. above 50% agreement must be achieved to actually add new shareholder.	Must	Arguments: address _newAcc, uint256 _shares Modifiers: public, payable Returns: bool (success
22	**calculateReputation()**		Must	

User interaction and design

Questions

Below is a list of questions to be addressed as a result of this requirements document:

Question	Outcome
How does PTR contract works?	In the deployment stage, an array of shareholders (accounts + shares) are provided to the constructor function, which in turn verifies the shareholders as registered entities and saves them to a variable.

Not Doing

-