**SOFTWARE REQUIREMENTS SPECIFICATION**

**FunixPricingChain**

**1. Product overview**

* Product name: FunixPricingChain – an online product pricing application.
* Product description: Create a platform where participants can suggest prices for a product, and at the end of each session, the administrator determines the final price. Participants who participate in the pricing process will accumulate a deviation score, which will be used as a weight in future sessions they join.
* Users – who will use the application:

|  |  |  |
| --- | --- | --- |
| *No.* | *User* | *Description* |
| *1.* | *Admin* | *The administrator is responsible for creating and overseeing the sessions, and ultimately making the final decision on the price for each product.* |
| *2.* | *Participants* | *The participants suggested prices for the products during each session on the platform.* |

* The benefit to users:

The FunixPricingChain Dapp is designed to address the problem of pricing for products with unknown prices. It is particularly beneficial for individuals who are unsure about the price of a product they want to buy or for products that do not have a specific price on the market.

By utilizing the FunixPricingChain Dapp, users can leverage the collective wisdom of the platform's participants to determine a fair and reasonable price for these products. The Dapp allows participants to suggest prices for the product, and their contributions play a crucial role in establishing the final price.

Through this decentralized approach, the FunixPricingChain Dapp provides a solution for pricing unknown price products, ensuring transparency and fairness in the pricing process.

* Functional Diagram:

A screenshot of a black screen

Description automatically generated

*Image 1: Functional diagram*

**2. Functional Requirements**

2.1 Create new session.

* Function name: createNewSession.
* Function description: The function is to initiate a new session for pricing product.
* User to use: Only admin will use the function.
* Function Inputs:

|  |  |
| --- | --- |
| **Input** | **Requirement** |
| Product name | String, required. |
| Product description | String, required. |
| Product images | List of string, each string refers to a specific image |
| Session duration | Uint256, required. |

* Function rule: System will require at least one image for each product. Images are stored on IPFS.
* Function result: a new product with valid information will be added and a pricing session is initiated.

A screenshot of a computer

Description automatically generated

*Image 2: Create new session form*

2.2 Participant to register.

* Function name: register.
* Function description: The function is to register new participant in a smart contract.
* User to use: Only users who not registered can use this function.
* Function Inputs:

|  |  |
| --- | --- |
| **Input** | **Requirement** |
| Participant name | String, required. |
| Participant email | String, required. |

* Function rule: Users must not have pre-registered and there can only be a maximum of 10 registrations.
* Function result: The user will be added to the user list for admin approval.
* Mockup screen:

A screenshot of a register

Description automatically generated

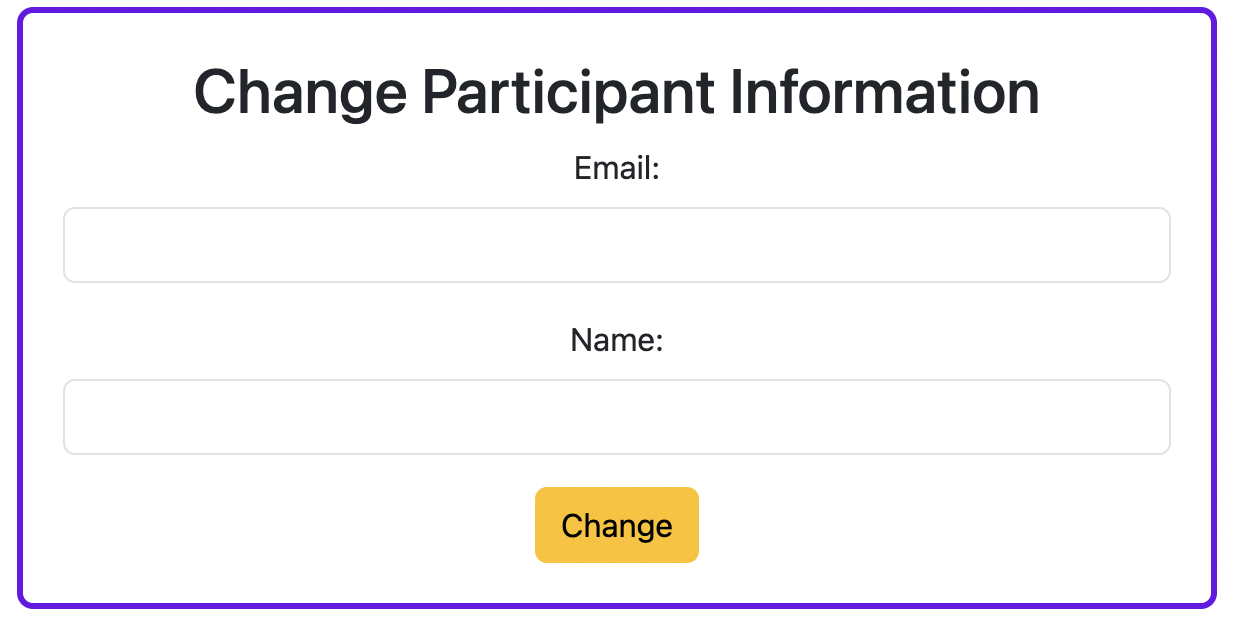
*Image 3: Register*

2.3 Change user information.

* Function name: changeParticipantInfo.
* Function description: The function is to change email and full name of user.
* User to use: Only users who already has registered.
* Function Inputs:

|  |  |
| --- | --- |
| **Input** | **Requirement** |
| Participant name | String, required. |
| Participant email | String, required. |

* Function rule: The system will need name or email or both to execute the function.
* Function result: Users will be able to change their name and email.
* Mockup screen:



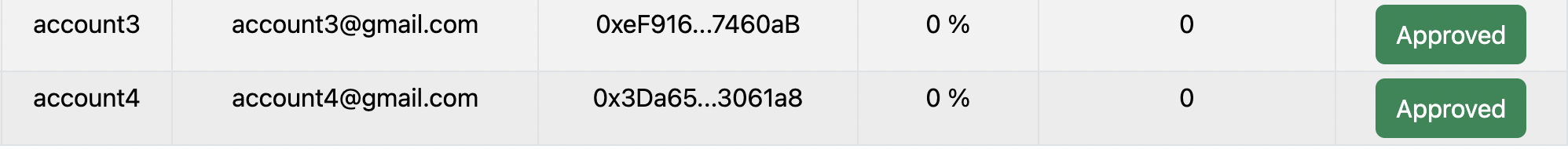
*Image 4: Change information*

2.4 Approve participants.

* Function name: approvedParticipant.
* Function description: The function is approved participants to have right to price the product.
* User to use: Only admin.
* Function Inputs:

|  |  |
| --- | --- |
| **Input** | **Requirement** |
| Participant address | Address, required. |

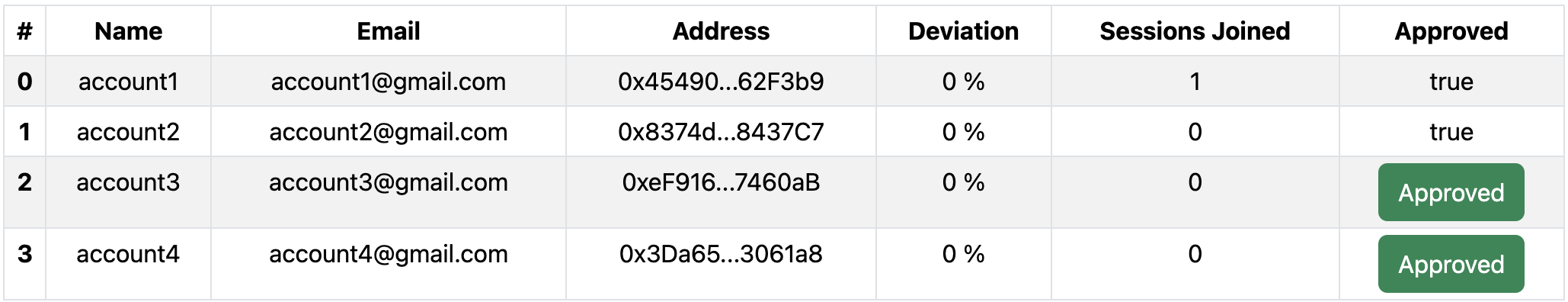
* Function rule: Must be registered before approval.
* Function result: People approved by admin can participate in the pricing process.
* Mockup screen:



*Image 5: Approved*

2.5 Get all users information.

* Function name: getParticipantList.
* Function description: displays the information panel of all users.
* User to use: Only admin.
* Function Inputs: This function has no input.
* Function rule: Only admin can use this function to get all users information.
* Function result: People approved by admin can participate in the pricing process.
* Mockup screen:



*Image 6: Users information*

2.6 Get all sessions detail.

* Function name: getSessionsInfo.
* Function description: The function is to allow participants and administrators to view all sessions that exist.
* User to use: Admin and users.
* Function Inputs: This function has no input.
* Function rule: Users registered and connected to the Dapp can view all sessions.
* Function result: Show all available sessions on screen for admin to end the sessions and participants to pricing products.
* Mockup screen:



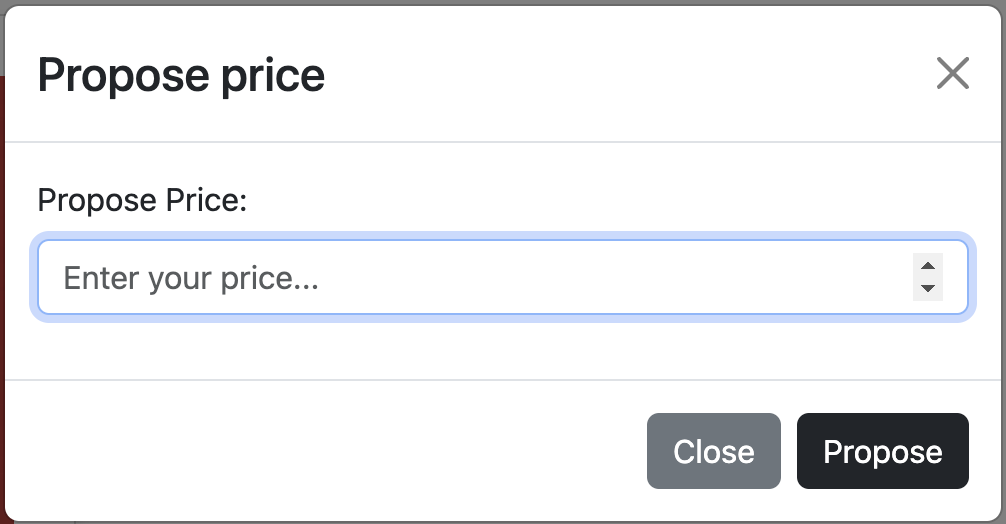
*Image 7: Sessions*

2.7 Participants propose products.

* Function name: proposeProductPrice.
* Function description: The function is allow participant to propose price for the product.
* User to use: Only approved users.
* Function Inputs:

|  |  |
| --- | --- |
| **Input** | **Requirement** |
| Propose price | uint256, required. |

* Function rule: Only participant who has already registered and approved can propose. And session must to still open.
* Function result: The participant's suggested price is stored to calculate the suggested price and update the number of sessions the participant has participated in one more time.
* Mockup screen:



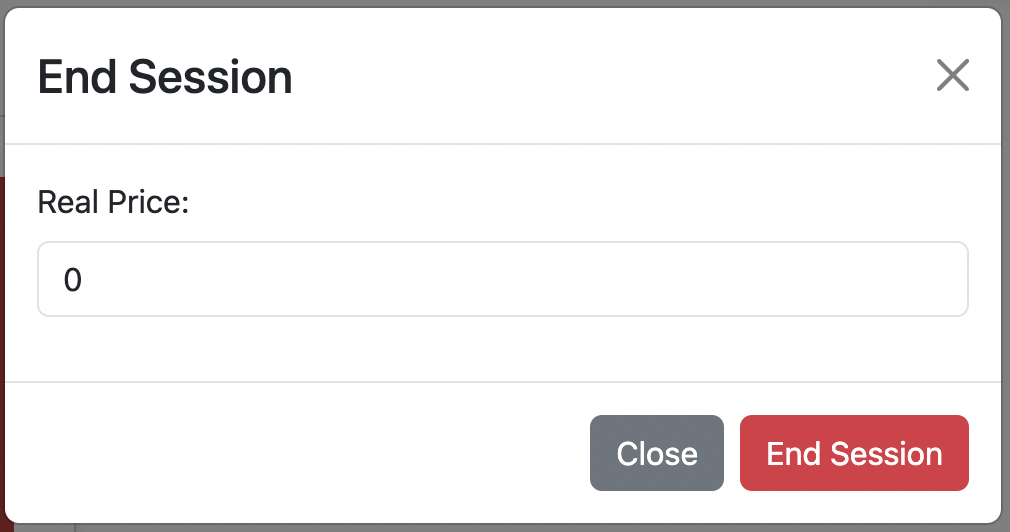
*Image 8: Propose price*

2.8 End session.

* Function name: endSession.
* Function description: Admin set a real price and end the session.
* User to use: Only admin.
* Function Inputs:

|  |  |
| --- | --- |
| **Input** | **Requirement** |
| Real price | Uint256, required. |

* Function rule: Only admin can use this function.
* Function result: The actual price and suggested price will be saved in the system, the state will switch to closed and the participant's deviation will be calculated to update.
* Mockup screen:



*Image 9: End Sessions*

**3. Non-Functional Requirements**

* Web browser coverage: Chrome, Microsoft edge, FireFox, Opera, … and browser that support Metamask.
* System performance: The system can work well with 10 people or less.

**4. Product Upgrades**

* Upgrades UX/UI*.*
* Add responsive for dapp frontend.
* Upgrades Dapp to compatibility Mobile platform.
* Active participants with small deviation can receive tokens.