SOFTWARE REQUIREMENTS SPECIFICATION

FunixPricingApplication

1. Product overview

* Product name: FunixPricingApplicaion – an online product pricing application.
* Product description:

It allows the participants to give their estimated price for producs. Participants will join and price for products based on their own experience and views . The final price will be set based on a calculation of all the given prices when admin stops the pricing session.

* Users – who will use the application:

|  |  |  |
| --- | --- | --- |
| *No.* | *User* | *Description* |
| *1.* | *Seller/Manufacturer* | *Have a demand to define price that for their products that meets the amount of money that customers willing to pay for* |
| *2.* | *Marketer* | *Have a demand to research about available price of a new product that will be brought to sell in the future* |

* The benefit to users:

*+ Sellers/Manufacturers are the key users that the application is targeting. The application can help sellers to know an average price that people willing pay for their product among others*

*+ Marketers are the key users that the product is targeting. The product can help them to know about available price of a new product that will be sold in future*

* Functional Diagram:

*The sample of the functional diagram is provided below*

*The details of each function will be specified in the following section.*

Admin

User

2. Functional Requirements

2.1 1. Initiate A Pricing Session

* Function name: Initiate A new Session
* Function description: the function is to initiate a new session of a product with name and description and id
* User to use: Only admin will use the function
* Function Inputs:

|  |  |
| --- | --- |
| Input | Requirement |
| Product name | String, required. |
| Product description | String, required. |

* Function rule: system will require to input both name and description of product. Id will be automatically created and be unique for each product. Only admin can edit information of product
* Function result: a new product with valid information will be added and a pricing session is initiated.

2.1 2. Participant to sign up/sign in

* Function name: register
* Function description: the function is for participants to sign up/sign in by MetaMask. After connect their account with Metamask they need to register name and email before joining any pricing session.
* User to use: Participants will use this function
* Function Inputs:

|  |  |
| --- | --- |
| Input | Requirement |
| Participant’s name | String, required. |
| Participant’s email | String with email structure, required. |

* Function rule: Both name and email are required inputs. Information of the participant will be maintained in the main Smart contract. Participants can edit your own email or name.
* Function result: a new user with valid information will be added .When the participant sign up and They can view their own detailed profile .

2.1 3. Participant to view and price a product

* Function name: View ongoing pricing sessions and give price for any product with unlimited pricing time. Only their final price for each session will be counted.
* Function description: View ongoing pricing sessions and give price for any product with unlimited pricing time. Only their final price for each session will be counted.
* User to use: participants will use this function
* Function Inputs:

|  |  |
| --- | --- |
| Input | Requirement |
| Price | uint, required. |

* Function rule:

- A participant can see all ongoing pricing sessions.

-  A participant can price a product more than once as long as the session is still open. The latest given price will be used

* Function result: A proposed price is calculated based on all given prices and participants’ deviation

2.1 3. Admin to view and close a pricing session

* Function name: View and close a pricing session
* Function description: the function is for admin to view and close a pricing session
* 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 |

* Function rule:

-      Admin can see all ongoing pricing sessions.

-      Admin can see details of the corresponding product

-      Admin can see the proposed price for each pricing session.

-      The formula of the proposed price is provided in project specifications.

-      Admin can close a pricing session. The proposed price will be stored on a smart contract.

Function result: -    A session is closed. The admin will update the final price. The state of pricing session is closed. When the final price is given, the individual deviation of all participants will be updated.

2.1 4. View a list of all participants with details

* Function name: View a list of all participants with details
* Function description: the function is for admin to view a list of all participants with details
* User to use: Only admin will use the function
* Function Inputs: None.
* Function rule: Data is automatically updated in table
* Function result: List of all participants with detailed information: name, email, number of joined sessions, accumulated deviation.

2.1 5. View a list of all sessions with details

* Function name: View a list of all sessions with details Session
* Function description: the function is for admin to view a list of all sessions with details
* User to use: Only admin will use the function
* Function Inputs: None
* Function rule: information need to be updated imediately when there is any change.
* Function result: List of all pricing sessions with detailed information: product info, state of the session, the proposed price, the final price.

2.1.6. Image of products will be stored in IPFS

* Function name: Image of products will be stored in IPFS
* Function description: the function is for admin to upload images from local storage to IPFS and show them for each session that is started
* User to use: Only admin will use the function
* Function Inputs: Images
* Function rule: Images stores in IPFS and to be store in smart contract as hashes in an array for each session
* Function result: Participants can see images of products when session is set to start.

Lay out home screen that participants see.

Sign up/Sign in

Product Image A

Product A Info

Price: