**SOFTWARE REQUIREMENTS SPECIFICATION**

**1. Product overview**

* Product name: FunixSwap – A decentralized exchange for swapping tokens.
* Product description:

+) FunixSwap performs as an exchange that facilitates trader to exchange ether to tokens, and between two distinct types of tokens.

+) Trader can also transfering tokens, ether from an account to another account or to a contract.

- The benefit to users:

+) FunixSwap performs as an exchange of ether and tokens. It provides connection between seller and buyer.

+) FunixSwap updates token rates and process transactions quickly. Customer are free from token sale scams, tokens and ether reach safely to buyers and sellers.

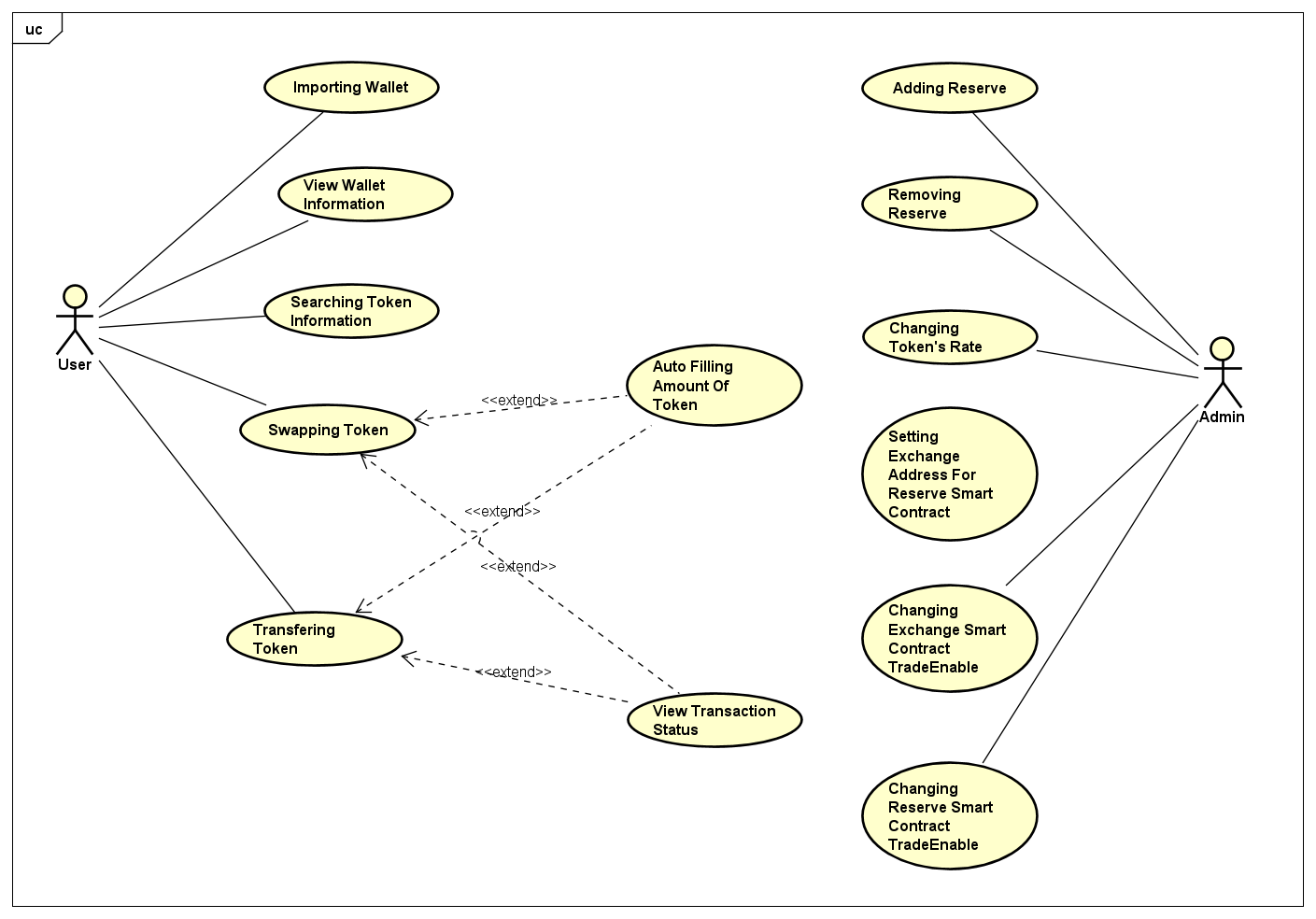
+) The transaction is carried out using a user-friendly interface.

* Actors:

|  |  |  |
| --- | --- | --- |
| No. | Actor | Description |
| 1. | User | The user who had a wallet on Ethereum blockchain. |
| 2. | Admin | The administrator of the exchange. |

**2. Functional Specifications**

**2.1. Use Cases Diagram**

**

**2.2. Use Cases Specifications**

**2.2.1. Importing Wallet.**

|  |  |
| --- | --- |
| **ID:** | UC1. |
| **Title:** | Importing Wallet. |
| **Description:** | User import wallet. |
| **Primary Actor:** | User. |
| **Preconditions:** | User is in homepage screen. |
| **Postconditions:** | Accounts was imported. |
| **Trigger:** | User clicks button “Metamask” in hompage screen. |
| **Main  Success Scenario:** | 1. User clicks button “Metamask” in hompage screen. 2. Metamask display popup for user choosing account to connect with. 3. User enter/select account to import. 4. User click button confirm to connect chosen accounts to FunixSwap. 5. Metamask connects accounts which have been chosen to FunixSwap’s Agent. |
| **Extensions:** | 1. Cancel choose wallet to be imported.  1. User click button canncel of popup of Metamask to choose accounts to be imported. 2. Return to HomePage. 3. User not loggined to metamask. 4. User clicks button “Metamask” in hompage screen. 5. Metamask display popup for user login to metamask. 6. User input login information. 7. Metamask login for user. 8. User enter/select account to import. 9. User click button confirm to connect chosen accounts to FunixSwap. 10. Metamask connects accounts which have been chosen to FunixSwap’s Agent. |

**2.2.2. View wallet information.**

|  |  |
| --- | --- |
| **ID:** | UC2. |
| **Title:** | View wallet information. |
| **Description:** | User view wallet information. |
| **Primary Actor:** | User. |
| **Preconditions:** | Wallet was imported. |
| **Postconditions:** | User balance displayed. |
| **Trigger:** | User opens interface to view wallet information. |
| **Main  Success Scenario:** | 1. User opens interface to view wallet information. 2. User agent displays address of current imported wallet. 3. User agent fetch balance of current account of all supported token. 4. User agent displays balance of each token. 5. User agent start background job to refresh user balance of all token for interval of 10 seconds. |
| **Extensions:** | N/A. |

**2.2.3. Searching token detail information.**

|  |  |
| --- | --- |
| **ID:** | UC3. |
| **Title:** | Searching token . |
| **Description:** | User searching token detail information |
| **Primary Actor:** | User. |
| **Preconditions:** | Wallet was imported. |
| **Postconditions:** | Token detail information displayed. |
| **Trigger:** | User opens interface to search token detail information. |
| **Main  Success Scenario:** | 1. User inputs token symbol or name and click button search. 2. User agent fetchs detail information of corresponding token. 3. User agent displays fetched token’s detail information. 4. User agent start background job to refresh data for interval of 10 seconds. |
| **Extensions:** | 1. User search for wrong token’s symbol or name. 2. User opens interface to search token detail information. 3. User input a wrong token’s symbol or name and click button search. 4. User agent fetchs detail information of corresponding token. 5. User agent receive no token’s detail information. 6. User agent show alert to user that the user is search for a non supported token. |

**2.2.4. Swapping Token.**

|  |  |
| --- | --- |
| **ID:** | UC4. |
| **Title:** | Swapping Token. |
| **Description:** | User Create Swapping Token Order. |
| **Primary Actor:** | User. |
| **Preconditions:** | Wallet was imported. User in swapping token screen. |
| **Postconditions:** | Swapping Token Order Was Processed. |
| **Trigger:** | User opens interface to create swapping token order. |
| **Main  Success Scenario:** | 1. User selects of source token type. 2. User inputs amount of source token. 3. User selects of destination token type. 4. User agent sends a query to Exchange to get amount of destination token will received. 5. User agent displays amount of destination token will received. 6. User clicks button “SWAP”. 7. User agent calculates transaction fee. 8. User agent opens popup to request user confirm the swapping tokens order. 9. User clicks button “Confirm”. 10. User agent close confirmation popup. 11. Metamask display popup to request user confirm and sign the transaction. 12. User Agent sign the transaction. 13. Metamask sends the transaction. 14. Exchange smart contract request corresponding Reserve smart contract to creates new sell order for source token type. 15. Exchange smart contract request corresponding Reserve smart conract to creates new buy order for destination token type. 16. Exchange smart contract request corresponding Reserve smart contract to performs a match of sell order and buy order of source token type. If there are matched orders, Reserve smart contract send ethers or tokens to buyer. 17. Exchange smart contract request corresponding Reserve smart contract to performs a match of sell order and buy order of destination token type. If there are matched orders, Reserve smart contract send ethers or tokens to buyer. |
| **Extensions:** | 1. User cancel in confirmation popup. 2. In the popup which user agent displays to request user confirm creating order, user click button “Cancel”. 3. Close the popup. 4. User reject to sign the transaction. 5. In the popup which Metamask display to request user signs to transaction, user click button “Cancel”. 6. Close the popup. 7. User have not approve enough token to create swapping token order. User does not have enough ether in his wallet. 8. In the popup which user agent display to request user confirm to create swapping token order. 9. User click button “Confirm”. 10. Exchange smart contract validate and see that user have not approve enough token to create swapping token order or he does not have enought ether in his wallet. 11. User agent display alert popup to notify user have not approve enough token to create swapping token order or he does not have enought ether in his wallet. |

**2.2.5. Transfering Token.**

|  |  |
| --- | --- |
| **ID:** | UC5. |
| **Title:** | Transfering Token. |
| **Description:** | User create transfer token order. |
| **Primary Actor:** | User. |
| **Preconditions:** | Wallet was imported. User in transfer token screen. |
| **Postconditions:** | Token transfered to receiver wallet. |
| **Trigger:** | User opens interface to create transfer token order. |
| **Main  Success Scenario:** | 1. User input address of receiver. 2. User selects token type to transfer. 3. User inputs amount of token to transfer. 4. User clicks button “TRANSFER”. 5. User Agent calculates Transaction Fee. 6. User Agent opens popup to request user confirm the order and transaction fee. 7. User clicks button “Confirm”. 8. User agent close confirmation popup. 9. Metamask display popup to request user confirm and sign the transaction. 10. User Agent sign the transaction. 11. Metamask sends the transaction. 12. Exchange smart contract send corresponding token to the receiver. |
| **Extensions:** | 1. User cancel in confirmation popup. 2. In the popup which user agent display to request user confirm creating order, user click button “Cancel”. 3. Close the popup. 4. User reject to sign the transaction. 5. In the popup which Metamask display to request user signs to transaction, user click button “Cancel”. 6. Close the popup. 7. User have not approve enough token to create transfering token order. User does not have enough ether in his wallet. 8. In the popup which user agent display to request user confirm to create swapping token order. 9. User click button “Confirm”. 10. Exchange smart contract validate and see that user have not approve enough token to create transfering token order or he does not have enought ether in his wallet. 11. User agent display alert popup to notify user have not approve enough token to create transfering token order or he does not have enought ether in his wallet. |

**2.2.6. Auto Fill Up Amount Of Token.**

|  |  |
| --- | --- |
| **ID:** | UC5. |
| **Title:** | Auto Fill Up Amount Of Token. |
| **Description:** | When creating swapping token order or creating transfering token order, user can use feature auto fill amount of source token to quickly specify amount of token to be swapped or transfered. |
| **Primary Actor:** | User. |
| **Preconditions:** | User selected type of token. |
| **Postconditions:** | A percentage of amount of current balance of the type of token filled to amount of token field. |
| **Trigger:** | User click on dropdown list to choose quickly filling amount of token options. |
| **Main  Success Scenario:** | 1. Options 25%, 50%, 75%, 100% display like drop-down list for user can choose. 2. User choose an option. 3. User Agent calculate corresponding amount of token, then fill to amount of token field. |
| **Extensions:** | N/A. |

**2.2.6. View Status Of Transaction.**

|  |  |
| --- | --- |
| **ID:** | UC6. |
| **Title:** | Monitoring Status Of Transaction. |
| **Description:** | When creating swapping token order or creating transfering token order, User Agent monitors and displays status of transaction to interface. |
| **Primary Actor:** | User Agent. |
| **Preconditions:** | An transaction have been sended by User Agent. |
| **Postconditions:** | Status of the transaction displayed. |
| **Trigger:** | Swapping or transfering transaction was send. |
| **Main  Success Scenario:** | 1. User agent open a popup for user monitor transaction status. 2. User Agent listen for event of the transaction. 3. User Agent received an event of the transaction. 4. User Agent update interface. 5. User Agent check if the transaction not ended then return to step 1. If the transaction ended then go to next step. 6. User agent close the popup. |
| **Extensions:** | N/A. |

**2.2.7. Changing token’s rate.**

|  |  |
| --- | --- |
| **ID:** | UC7. |
| **Title:** | Changing token’s rate. |
| **Description:** | Admin changing token’s rate. |
| **Primary Actor:** | Admin. |
| **Preconditions:** | N/A. |
| **Postconditions:** | Token’s rate in reserve changed. |
| **Trigger:** | Admin send transaction to Exchange smart contract to change token’s rate. |
| **Main  Success Scenario:** | 1. Exchange smart contract request corresponding Reserve smart contract to update rate. 2. Exchange smart contract request all Reserve to loopup and update related swapping order. |
| **Extensions:** | N/A. |

**2.2.8. Adding reserve.**

|  |  |
| --- | --- |
| **ID:** | UC8. |
| **Title:** | Adding reserve. |
| **Description:** | Admin add new reserve to exchange. |
| **Primary Actor:** | Admin. |
| **Preconditions:** | N/A. |
| **Postconditions:** | New reserve added to exchange |
| **Trigger:** | Admin send transaction to add new reserve to exchange. |
| **Main  Success Scenario:** | 1. Exchange add new reserve to its list of supported reserves. |
| **Extensions:** | 1. Token has existed in exchange’s list of supported token. 2. Exchange check havetoken of the reserve existed in list of supported tokens. 3. The token of the reserve has existed in list of supported tokens. 4. Revert the transaction. 5. User is not the owner of exchange smart contract. 6. Revert the transaction. |

**2.2.9. Adding reserve.**

|  |  |
| --- | --- |
| **ID:** | UC9. |
| **Title:** | Remove reserve. |
| **Description:** | Admin remove a reserve from exchange. |
| **Primary Actor:** | Admin. |
| **Preconditions:** | N/A. |
| **Postconditions:** | Corresponding reserve removed from exchange’s list of supported reserves. |
| **Trigger:** | Admin send transaction to remove a reserve from exchange. |
| **Main  Success Scenario:** | 1. Exchange remove the reserve from its list of supported reserves. |
| **Extensions:** | 1. User is not the owner of exchange smart contract. 2. Revert the transaction. |

**2.2.10. Setting Exchange Address For Reserve Smart Contract.**

|  |  |
| --- | --- |
| **ID:** | UC10. |
| **Title:** | Setting Exchange Address For Reserve Smart Contract |
| **Description:** | Admin set exchange address information of a reserve (assign a reserve to an exchange, then only the exchange can control the reserve). |
| **Primary Actor:** | Admin. |
| **Preconditions:** | N/A. |
| **Postconditions:** | Exchange address information of the reserve was setted. |
| **Trigger:** | Admin send transaction to change exchange address information of a reserve. |
| **Main  Success Scenario:** | 1. Reserve set the address to its exchange address. |
| **Extensions:** | 1. User is not the owner of the reserve smart contract. 2. Revert the transaction. |

**2.2.11. Changing Exchange Smart Contract TradeEnable.**

|  |  |
| --- | --- |
| **ID:** | UC11. |
| **Title:** | Changing Exchange Smart Contract TradeEnable |
| **Description:** | Admin sets ‘tradeEnable’ flag state of exchange smart contract. |
| **Primary Actor:** | Admin. |
| **Preconditions:** | N/A. |
| **Postconditions:** | State of ‘tradeEnable’ flag of the exchange smart contract was setted. |
| **Trigger:** | Admin send transaction to set ‘tradeEnable’ flag state of exchange smart contract. |
| **Main  Success Scenario:** | 1. The exchange smart contract set the ‘tradeEnable’ flag to the data that Admin request. |
| **Extensions:** | 1. User is not the owner of the exchange smart contract. 2. Revert the transaction. |

**2.2.12. Changing Reserve Smart Contract TradeEnable.**

|  |  |
| --- | --- |
| **ID:** | UC12. |
| **Title:** | Changing Reserve Smart Contract TradeEnable |
| **Description:** | Admin sets ‘tradeEnable’ flag state of reserve smart contract. |
| **Primary Actor:** | Admin. |
| **Preconditions:** | N/A. |
| **Postconditions:** | State of ‘tradeEnable’ flag of the reserve smart contract was setted. |
| **Trigger:** | Admin send transaction to set ‘tradeEnable’ flag state of a reserve smart contract. |
| **Main  Success Scenario:** | 1. The exchange smart contract request corresponding reserve smart contract to change its ‘tradeEnable’ flag state to the state that Admin requested. 2. The reserve smart contract set the ‘tradeEnable’ flag to the data that Admin request. |
| **Extensions:** | 1. User is not the owner of the exchange smart contract. 2. Revert the transaction. 3. The exchange smart contract’s address have not setted to the exchange address information of the reserve smart contract. 4. Revert the transaction. |

**3.Non-Functional Requirements**

* Web browser coverage: Chrome, Firefox.
* Number of user can access exchange at the same time: 200.

**4.Product Upgrades**

- The below functions will be added to the product soon:

+) Licensing user to withdraw the remain source token in created swapping token order.

+) Support Android OS and iOS.