Blockchain & Solidity Project

Project Questions:

1. Why will this project be important to Fintech and Blockchain/DeFi?
    a.
2. What problem are we working to solve?
    a. Cutting out the middle man (real estate broker). Facilitating person to person transactions in the field of Real Estate.
3. What data can be provided?
    a.
4. What methods, processes, functionalities are being chosen and why?
    a. We elected to use blockchain because it democratizes the process of transacting real estate.
5. What are some of the most important features and why?
    a. The ability to buy and sell NFTs (houses). The ability to transact for those NFTs with our proprietary fungible token (MRE)
6. What processes will we try or have been tried? (Question for a later not)
7. What technologies will be use?
    a. Solidity, Ganache, Streamlit, web3, python

Broad Functionality Overview:

Meta RE

Fungible
- ERC-20
- Pay 1ETH : 1 MRE(MetaRealEstate)

Non-Fungible
- ERC-721 (RET-RealEstateToken)
- Input Address
- Tied to the Wallet address

```
import
"https://github.com/OpenZeppelin/openzeppelin-contracts/blob/release-v2.5.0/contracts/token/E
RC721/ERC721Full.sol";

contract RealEstateToken is ERC721Full {
   address public admin;
```

```
    enum TokenStatus { Pending, Approved, Rejected }
    mapping(uint256 => TokenStatus) public tokenStatuses;

    constructor() public ERC721Full("MetaRE", "MRE") {
        admin = msg.sender;
    }

    modifier onlyAdmin() {
        require(msg.sender == admin, "Only the admin can call this function");
        _;
    }

    function setTokenStatus(uint256 tokenId, TokenStatus status) external onlyAdmin {
        require(tokenId < totalSupply(), "Invalid token ID");
        tokenStatuses[tokenId] = status;
    }

    constructor() public ERC721Full("MetaRE", "MRE") {
        admin = msg.sender;
    }

    function registerHouse(string memory tokenURI)
        public
        returns (uint256)
    {
        require(tokenStatuses[tokenId] == TokenStatus.Approved, "Token must be approved for
minting");

        uint256 tokenId = totalSupply();
        _mint(msg.sender, tokenId);
        _setTokenURI(tokenId, tokenURI);

        return tokenId;
    }
}
```

Verification

- Dummy function which takes in the user’s application for a token and returns token
- Users house address must be unique
- When new property is placed on the blockchain, documentation (deed) should be verified by parties [who have verified their identity]

List
- Function to decide whether you are willing to list a property for sale
- In order to list, must own token you are listing

Buy
- Token you want to buy has to be marked as “For Sale”
- Token has to be a valid address
- MRE wallet balance > MRE Home Listing Price

Sale

Terms and conditions after purchase or sale [some sort of button that acts as a proxy for a legal document, the users just have to click it]

Application Processes/Steps:

1. Create MetaRE contract
    a. Create contracts for ERC-20 and ERC-755
2.

Streamlit Design

1. Wallet Address (REQUIRED)
2. Wallet Balance
3. Current Portfolio of Properties
    a. Sell Button Options
        - Input the the price you want to sell at

4. BUY
    a. List of Properties for Sale
        - A list of available properties for sale
        - Buy only if wallet balance higher than listing

https://streamlit.io/gallery

Project Readme Overview

(For Readme)
Project Description…
Business Case - What problem is this trying to solve?

Software - Streamlit, Solidity, Python

Current Process as of 08/17/23:
Register house
Receive token URI
Revceive token pending until contract owner approve
The can add token/house for sale

Lead to mint, check status, wallet balance at the top, buy, portfolio

Question: Why am I spending that money?
Things done out of the solidity contract makes things less secure

Line 22: register house
Line 6: get token status
Line 33: add token for sale\
Line 90: get user tokens
    Token URI will show address
Line 40: get tokens for sale = Browse page