CRYPTO FLEA MARKET

By Vivian Shen

Yau H Chan

Isehise Ajayi

EXECUTIVE SUMMARY

- To create a decentralized exchange platform to swap Ether with USDC. Customers can set prices (limit order) for ETH and form up a contract and deal with the counterparty customer directly. A wallet address is the only information needed for setting up new customers. The benefits of setting up an exchange like this is to protect customers' privacy, customization of prices for different customers' needs and fast settlements.

OBJECTIVE

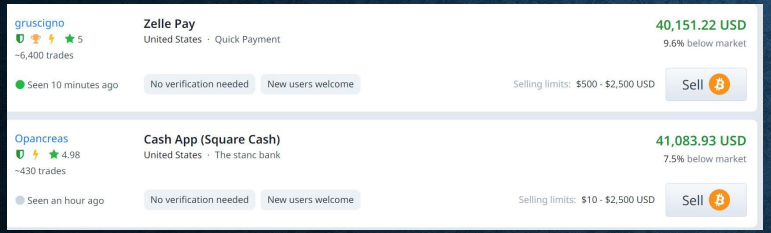
To create a Cryptocurrency Exchange Platform that is low-cost, trustworthy, and keep user's private information safe.

Primarily, we want to trade USDC with Ether. Buyers and Sellers can post their offer on the user interface. Viewers can select their offers and fill them.

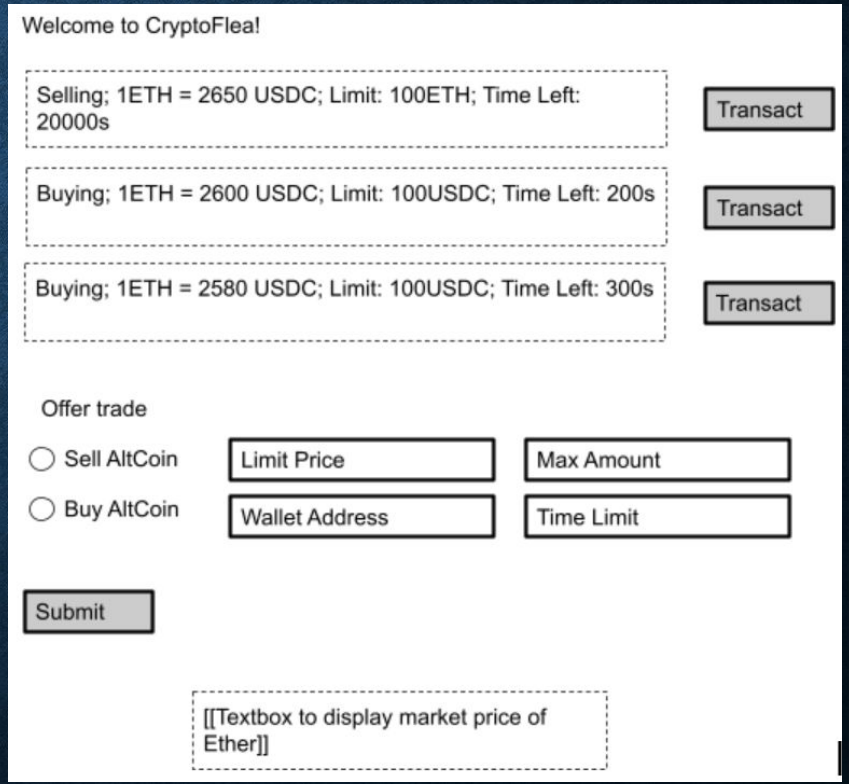
BENEFITS OF THE CRYPTO FLEA MARKET

- Create Arbitrage opportunities
- Flexibility and customization on orders.
- Increase Liquidity
- Avoids the huge spread of ether e.g wealth simple
- Privacy

User interface Design



Interface of localcoinswap

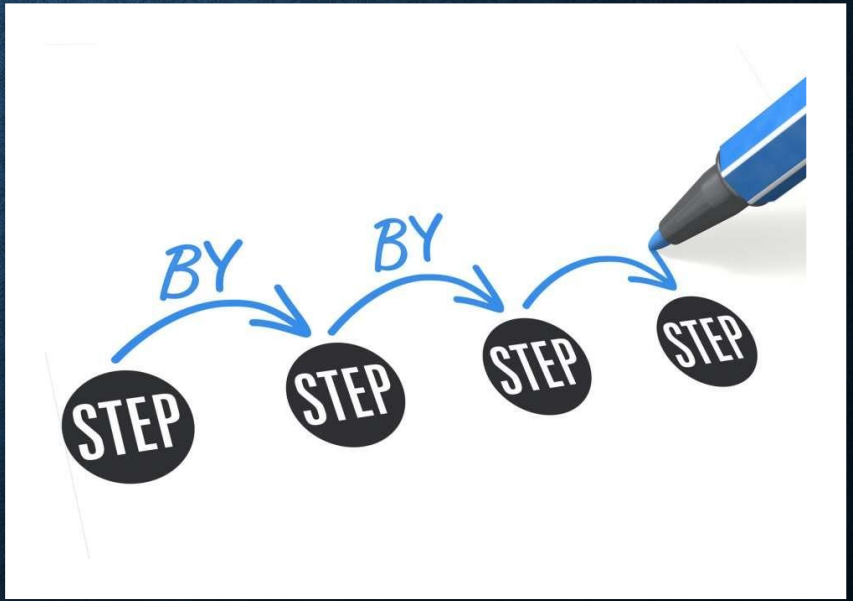


FRONT END INTEGRATION

- Python + Streamlit is used to create a User Interface, Streamlit Session State API is implemented to make the interface interactive

Step for integration:

- User Interface design is first sketched and coded without the backend engine
- To ensure robustness and testing, a simple Streamlit application is created to test the interaction between solidity engine and the user interface
- After the functions are tested thoroughly, the solidity functions are added to the user interface.



BACK-END PROCESS

1. How to get some USDC tokens for testing and presentation?

- USD Coins are formed by smart contracts on Ethereum.
- It’s fungible token, thus follow ERC20.
- Smallest unit is 6 decimal places.

USDC_testnet.sol

```
contract UsdcToken is ERC20 {
    address payable owner;
    bool public transferable = true;
    modifier onlyOwner {
        require(msg.sender == owner, "You do not have permission to mint these tokens!");
        _;
    }

    modifier istransferable {
        require(transferable==true, "token under lock, can Not Trade");
        _;
    }

    constructor() ERC20("USDCtest", "USDC")  {
        owner = payable(msg.sender);
        //default initial supply of the token is 10000USDC (6 decimals)
        _mint(owner, 10000000000);

    }
```

BACK-END PROCESS

2. How the swap works? How to lock the amount so to make initializer commit?

Initializer

Settler

`contract Dex_ETH_owner_trading`

- Owns ETH, wants USDC
- Owns USDC, wants ETH

```
function settleSwap (address payable settler,address payable _token, uint USDC_amount)
    external
    inState(State.Locked) OnlyOwner condition(msg.value != 0)
    payable {
        emit SettleSwap();
        state = State.Release;
        token = IUsdcToken(_token);
        settler.transfer(msg.value);
        initializer.transfer(msg.value);
        _safeTransferFrom(token,settler,initializer, USDC_amount);
    }
```

`contract Dex_USDC_owner_trading`

- Owns USDC, wants ETH
- Owns ETH, wants USDC

```
function settleSwap (uint init_amount, address payable settler_,address _token)
    external
    inState(State.Locked)
    condition(init_amount != 0)
    condition(msg.value !=0)
    payable {
        emit SettleSwap();
        token = IUsdcToken(_token);
        settler=settler_;
        require (msg.sender==settler, "please switch to settler's account for settling this swap");
        require (settler.balance >= msg.value,"settler, please make sure you have enough ETH amounts to swap");
        token.isTransferable(true);
        state = State.Release;
        _safeTransferFrom(token, initializer,settler,init_amount);
        initializer.transfer(msg.value);
    }
```

Further Enhancement in the Future

- Introduce withdraw function on Front end
- Market price displaying on the Front end page

Thank you!
Demo and Q&A time!

- More information on this project @ https://github.com/VvnShen/Project3/blob/main/README.md