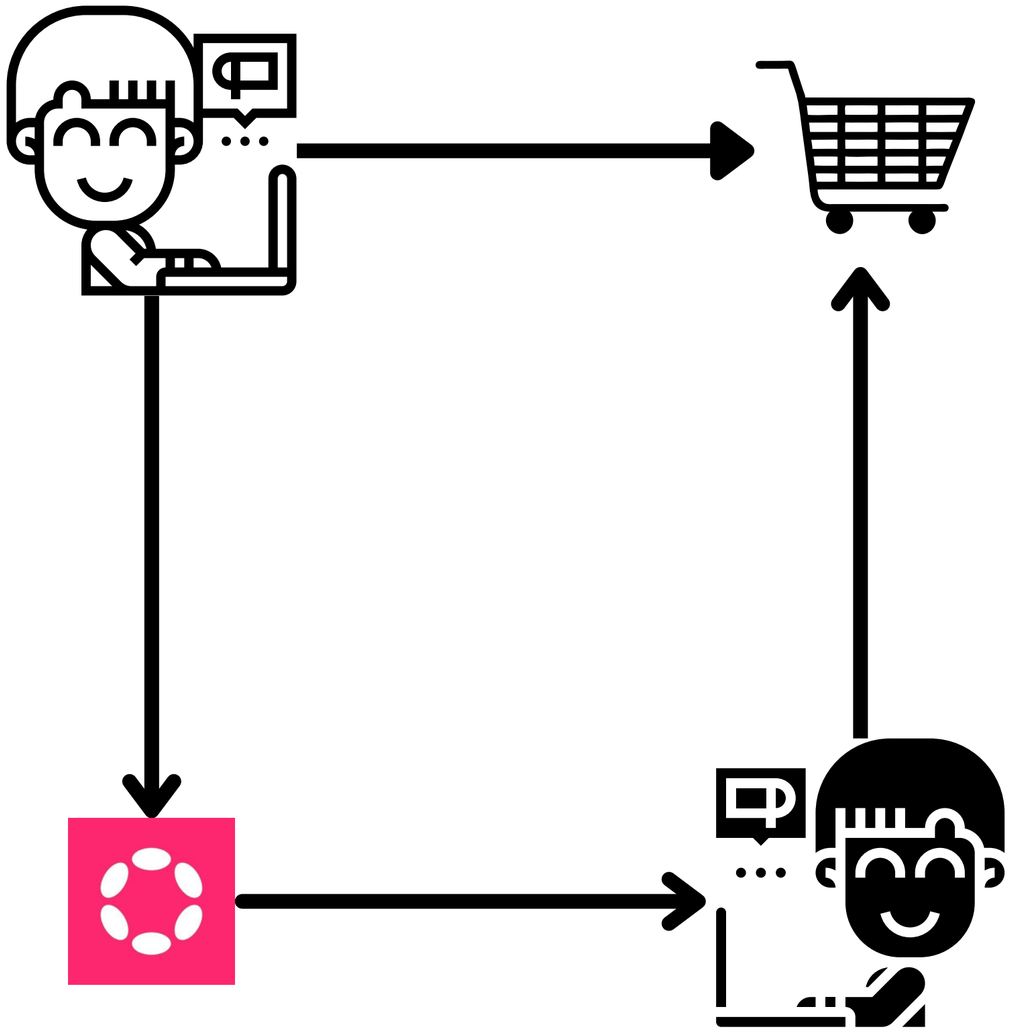
Decentralized Open Free Trade Platform

Index

01

Project Introduction

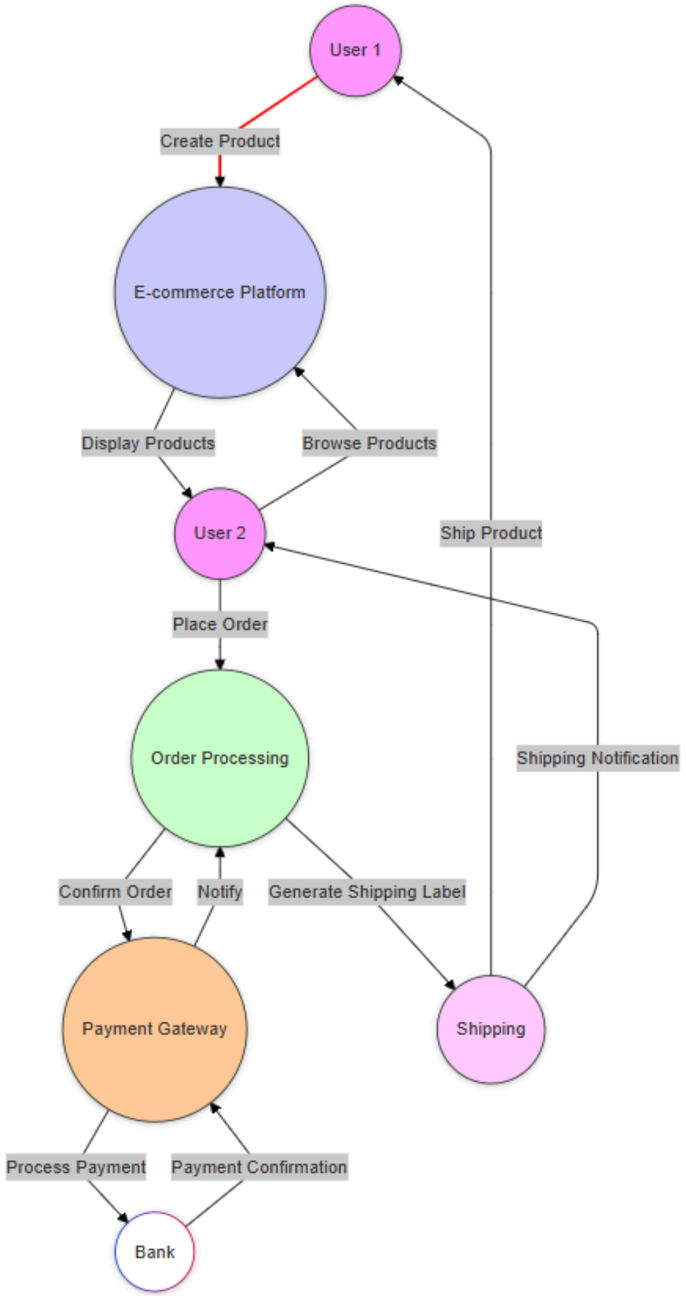


LibMarket aims to explore the possibility of decentralized e-commerce platforms, realize a Defi platform based on the C2C model for free transactions, and provide a free trading market system. Buyers and sellers can communicate directly and securely. Different from the current traditional trading model of "sellers set prices, buyers seek to buy NFTs", we negotiate off-chain platform-secured transactions on the chain for all off-chain items, including physical objects, without the need for third-party trust. This realizes a new market model where buyers and sellers can communicate and set prices.

02

Current market issues

Traditional architecture



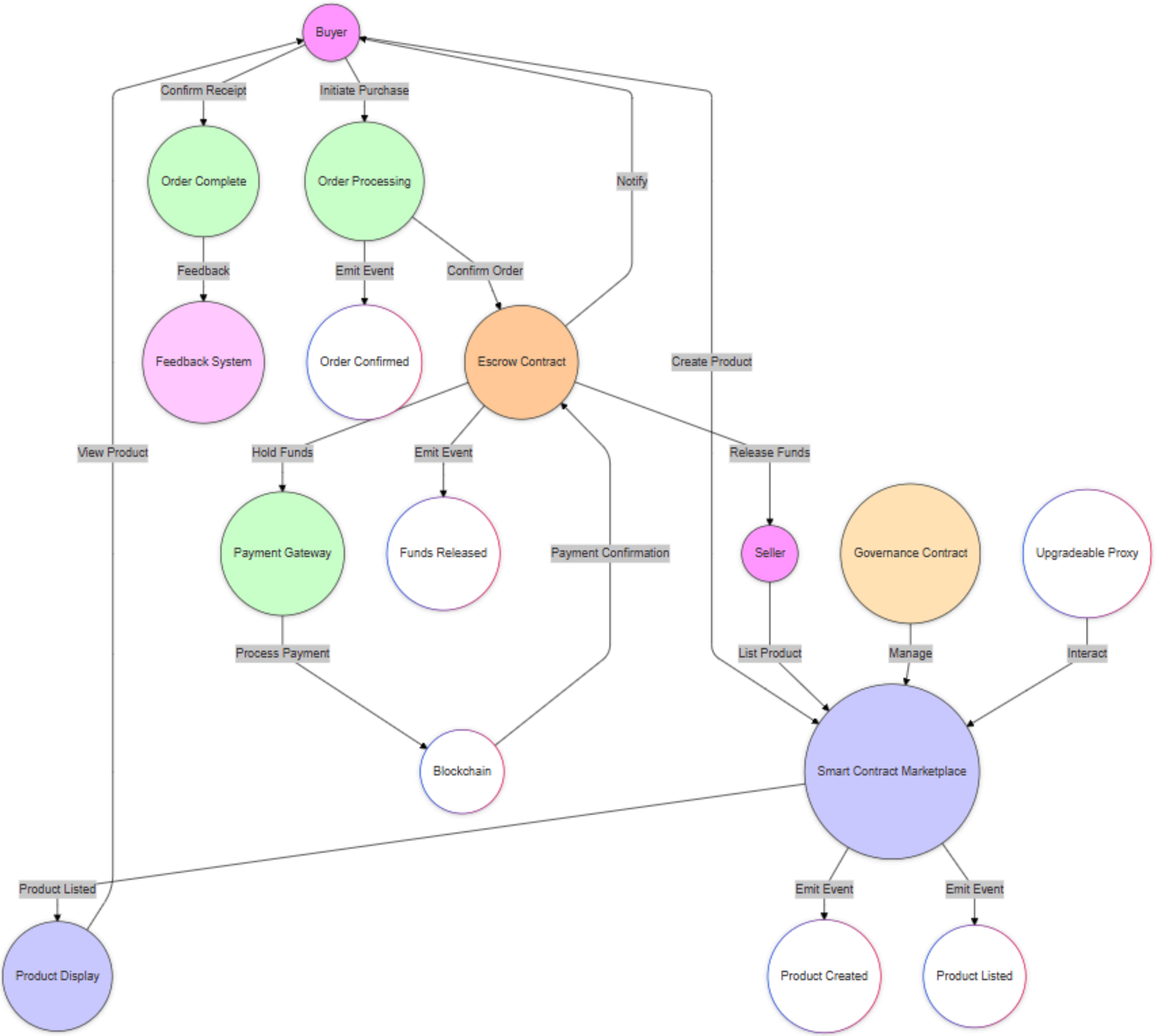
Advantages:

Community driven: Buyers and sellers establish connections through the platform, forming a good community atmosphere.
Flexibility: With third-party platform endorsement, after-sales and customer service processing.

Disadvantages:

Trust issues: Lack of trust between buyers and sellers, prone to fraud.
Difficult quality control: It is difficult for the platform to supervise the quality of all goods, affecting user experience.
Complex dispute handling: Handling sales disputes and complaints requires a lot of human resources.

Smart Contract Model Architecture



Advantages:

Transparency: Smart contracts are open on the blockchain, all transaction records are traceable, and trust is enhanced.
Automation: Transactions and payments are automatically executed through smart contracts, reducing manual intervention and improving efficiency.
Decentralization: Without the intervention of a centralized platform, transactions between users are more direct.
Security: The use of blockchain technology reduces the risk of fraud and data tampering.

Intermediate:

Order processing: There may be a need for after-sales service, when trust mechanisms such as points can be introduced

Disadvantages:

Network congestion: When the transaction volume is high, the blockchain network may be congested, resulting in transaction delays.

Feature category	eBay	Goofish	Lazada	LibMarket
Decentralized mechanism	✗	✗	✗	✓
Product Performance	middle	middle	low	high
Privacy and Security	middle	low	middle	**high**
Scalable functionality	low	middle	middle	**high**
barrier to entry	高	high	high	**low**
Cross chain asset trading	✗	✗	✗	✓
operating costs	high	middle	high	**low**
Trust mechanism	Relying on user feedback and platform reputation			**Establishing trust through smart contracts and blockchain transparency**
Platform control	Centralized platform management			Decentralization, user initiated transactions
transaction efficiency	More manual intervention and slower transaction speed			Automated execution, fast trading
Dispute resolution	Artificial intervention, handling complexity			Smart contract automatic execution reduces disputes

03

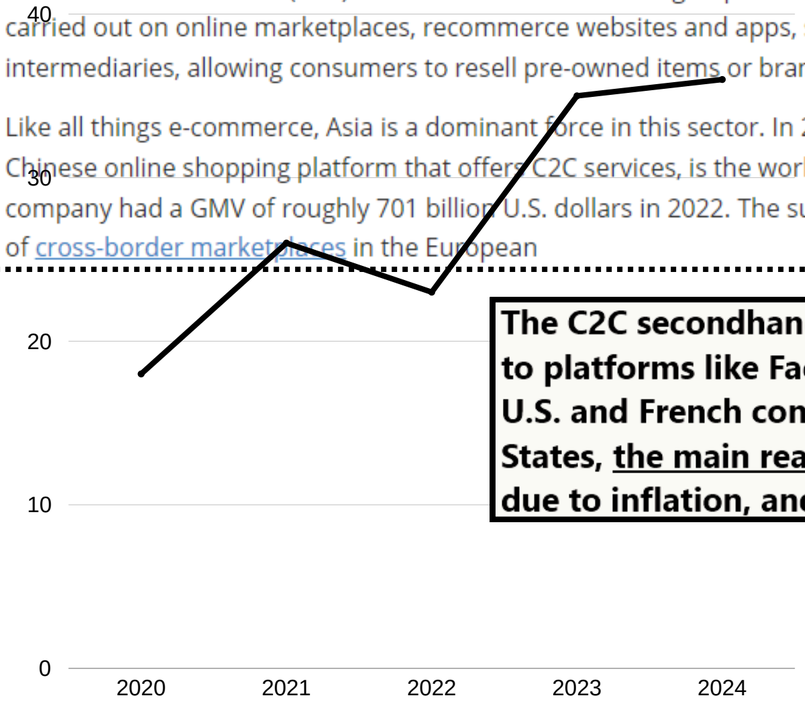
Development potential

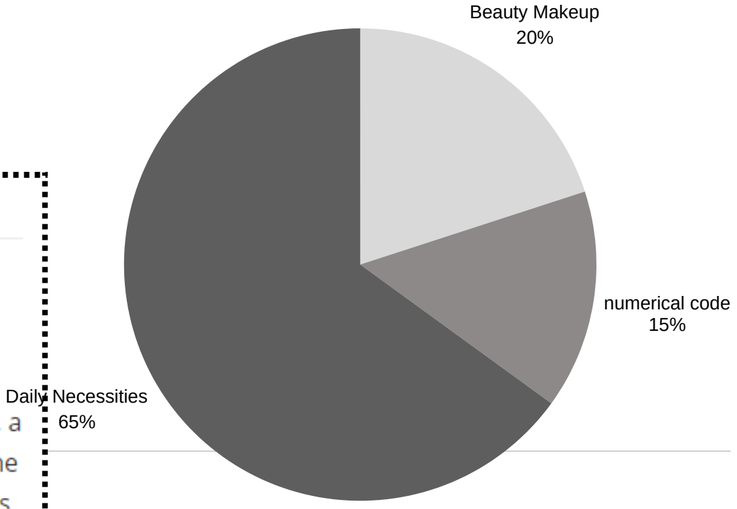
Traditional C2C market situation

C2C e-commerce - statistics and facts

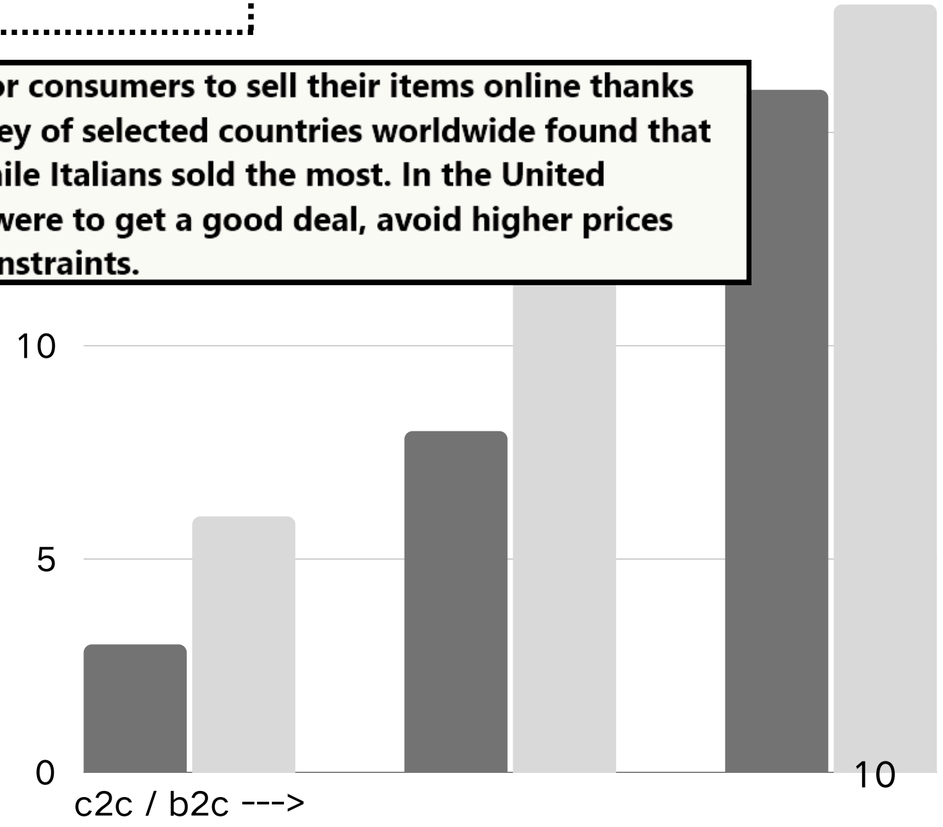
Consumer-to-consumer (C2C) e-commerce refers to selling or purchasing goods online between two private users. The C2C business model can be carried out on online marketplaces, recommerce websites and apps, sharing services, auction sites, and classifieds websites. The platforms act as intermediaries, allowing consumers to resell pre-owned items or brand-new goods.

Like all things e-commerce, Asia is a dominant force in this sector. In 2022, the C2C market accounted for one-fifth of China's online retail sales. Taobao, a Chinese online shopping platform that offers C2C services, is the world's most popular online marketplace based on gross merchandise value (GMV). The company had a GMV of roughly 701 billion U.S. dollars in 2022. The success of C2C trade can also be observed in Europe. That same year, over two-fifths of cross-border marketplaces in the European





The C2C secondhand market has become a very convenient place for consumers to sell their items online thanks to platforms like Facebook Marketplace, Depop, and Vinted. A survey of selected countries worldwide found that U.S. and French consumers purchased the most pre-loved items, while Italians sold the most. In the United States, the main reasons for buying and selling used items in 2023 were to get a good deal, avoid higher prices due to inflation, and maintain a desired lifestyle without budget constraints.



10

Web3 c2c model open trade platform potential

3%↗

30 B+

3 billion+

Annual growth rate 3%

Amazon + eBay
The entire market share
exceeds 60%

User Scale

Ecological Effect

Head Effect

Enhance transparency

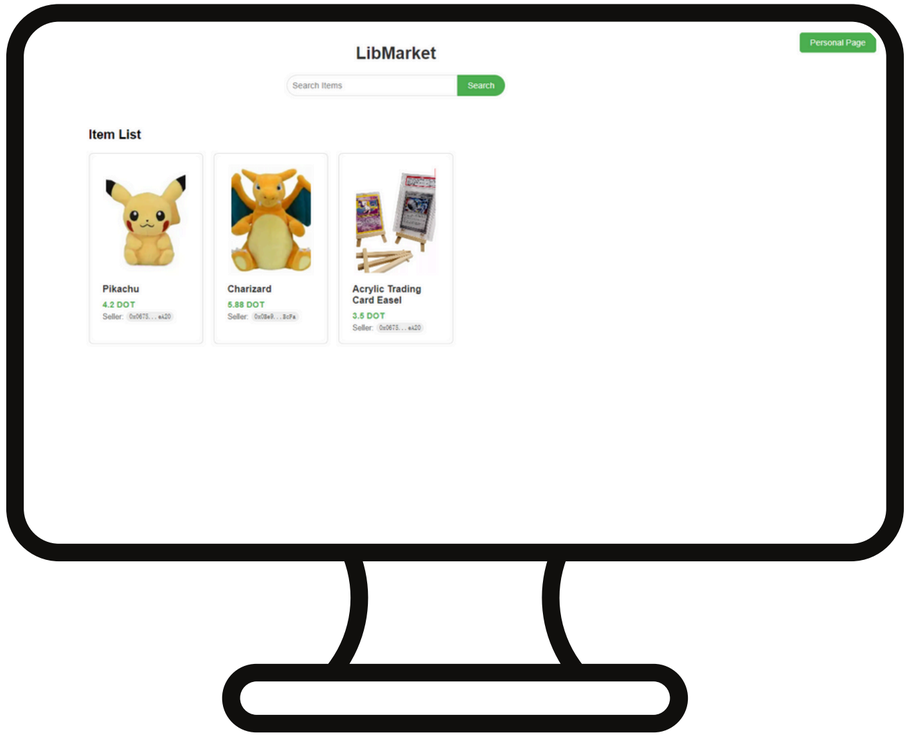
Reduce transaction costs

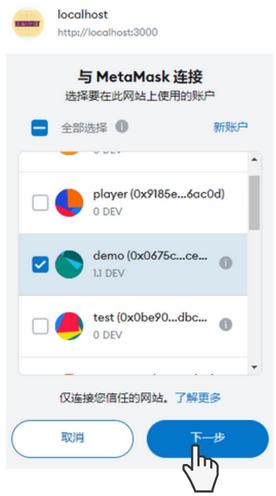
Improve accessibility of financial services

04

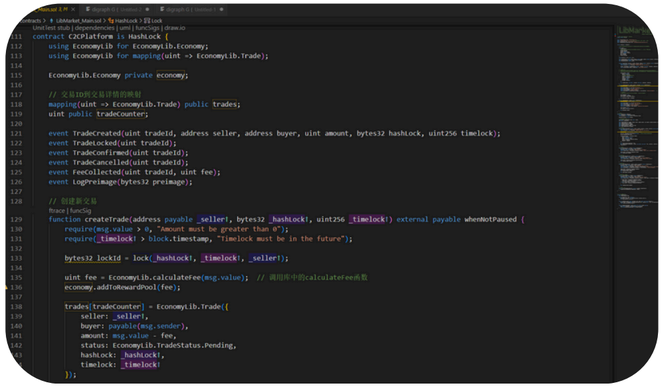
Technical Architecture

Front-end and back-end interaction architecture



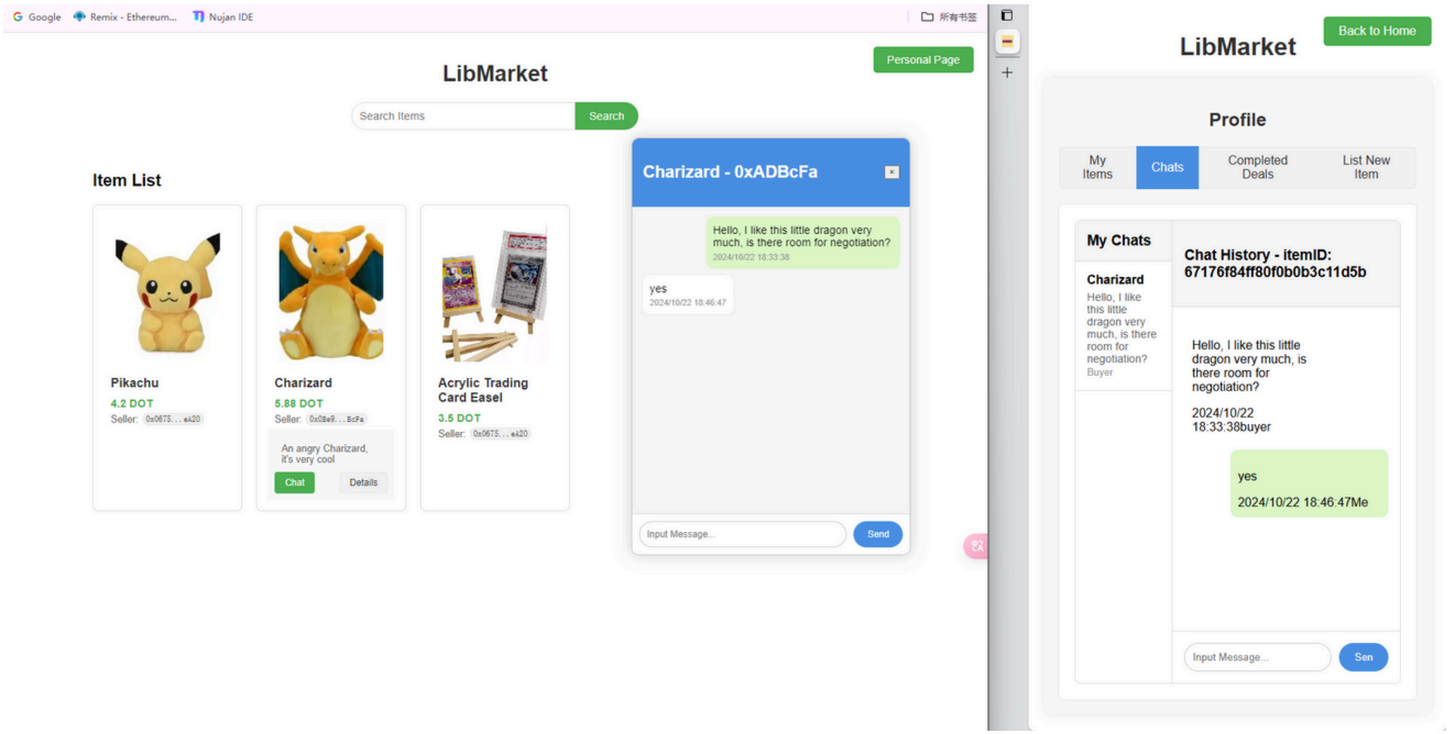


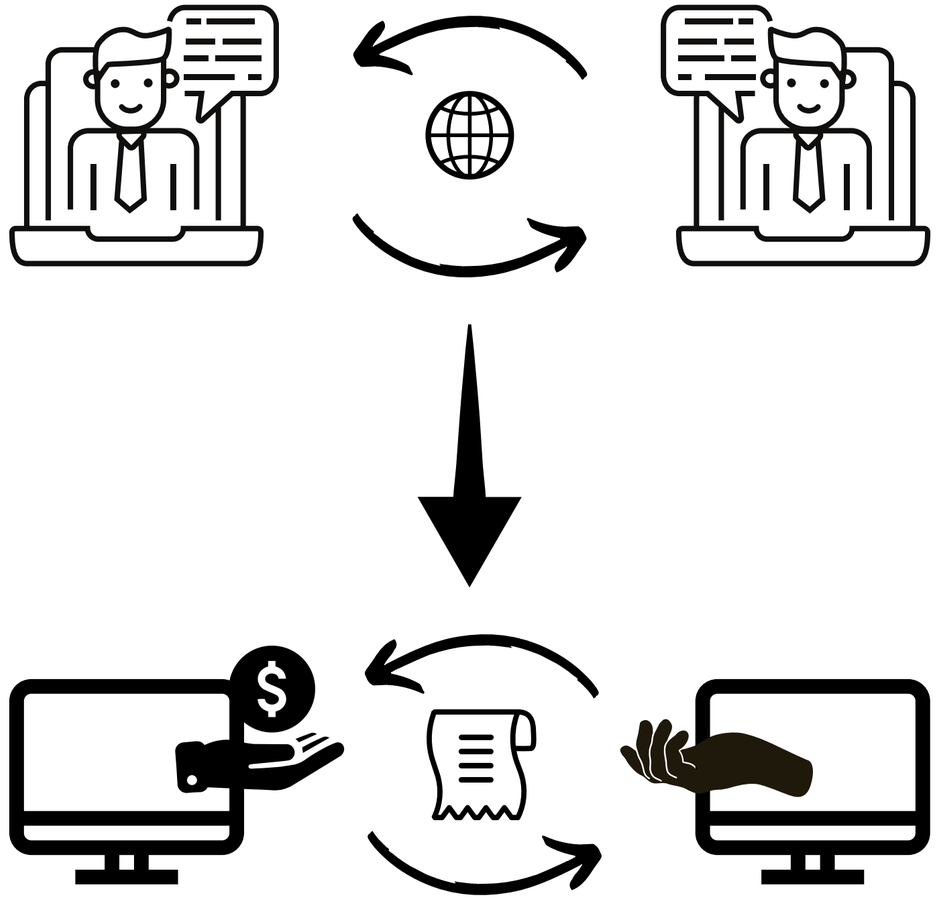
```
import Web3 from 'web3';

let LibMarketMainABI;

async function loadABI() {
  const response = await fetch('https://d.cess.network/870781361.json')
  LibMarketMainABI = await response.json();
}

loadABI();

const contractAddress = '0x7e2955a538d2b396cdfd172fd150c24b46d70cee';
const moonbaseAlphaRPC = 'https://moonbase-rpc.dwellir.com';
```



Function Realization

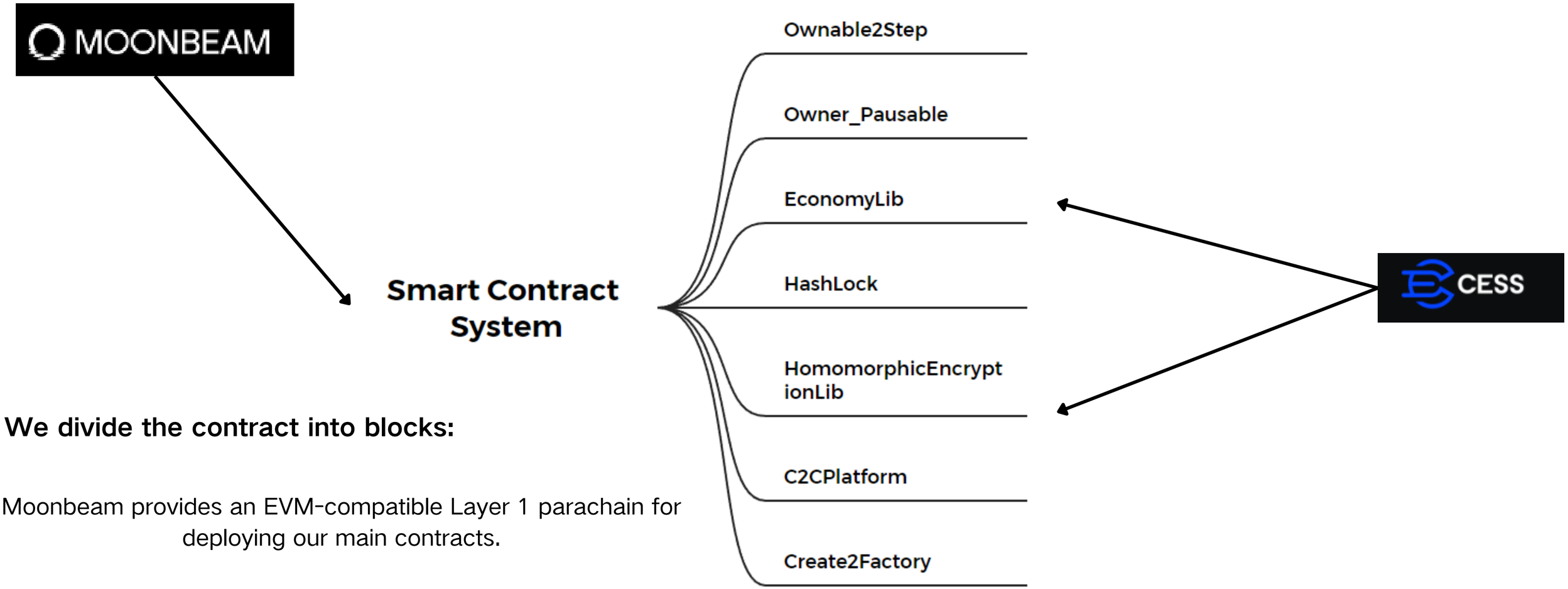
The buyer and seller can use instant communication to understand the product details and confirm the final price.





Smart Contract Architecture

https://moonbeam-explorer.netlify.app/tx/0xfe5bab25bf7db3f5b5d7f7db89a6536d93f0d58d274f27384f62b85e156bb3fb?network=MoonbaseAlpha

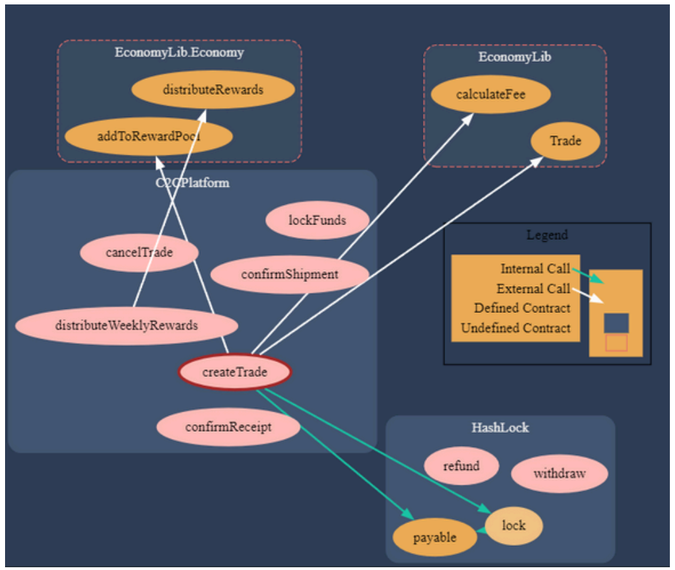


We divide the contract into blocks:

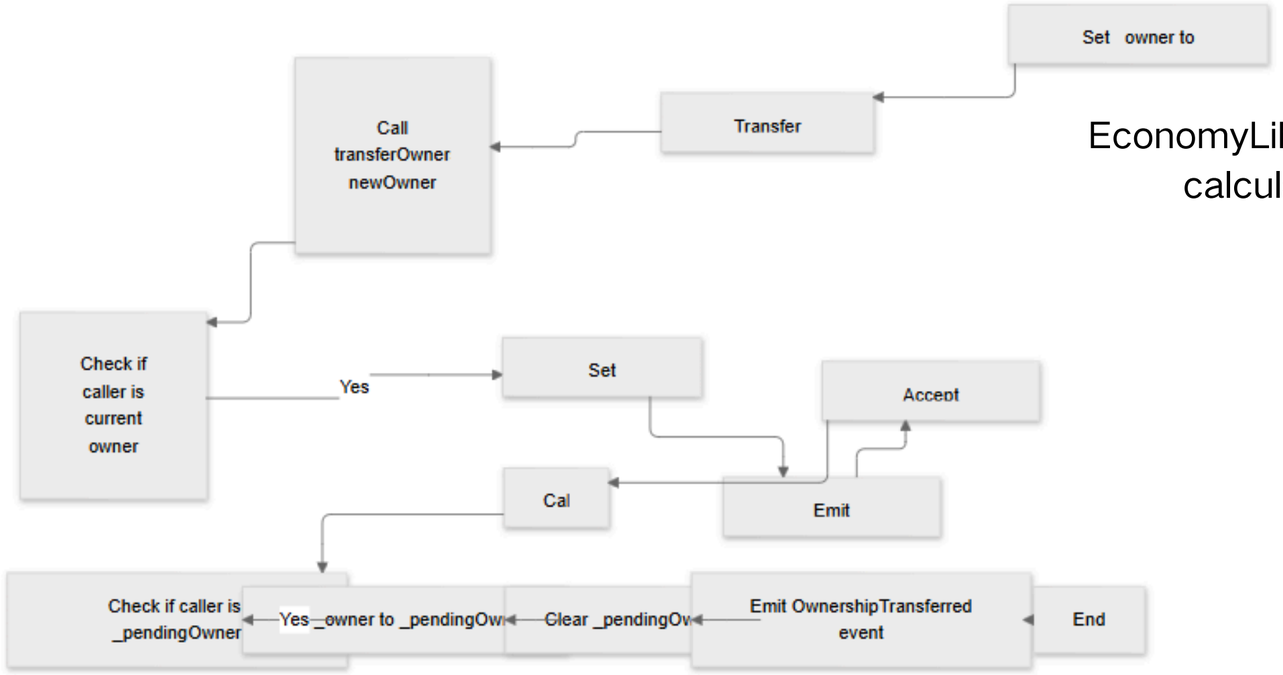
Moonbeam provides an EVM-compatible Layer 1 parachain for deploying our main contracts.

At the same time, we use CESS as data storage dependency and zk-proof calculation. CESS provides the best solution for storing and retrieving high-frequency dynamic data, facilitating information interaction between LibMarket users and rapid response to listing items.

Smart contract full architecture

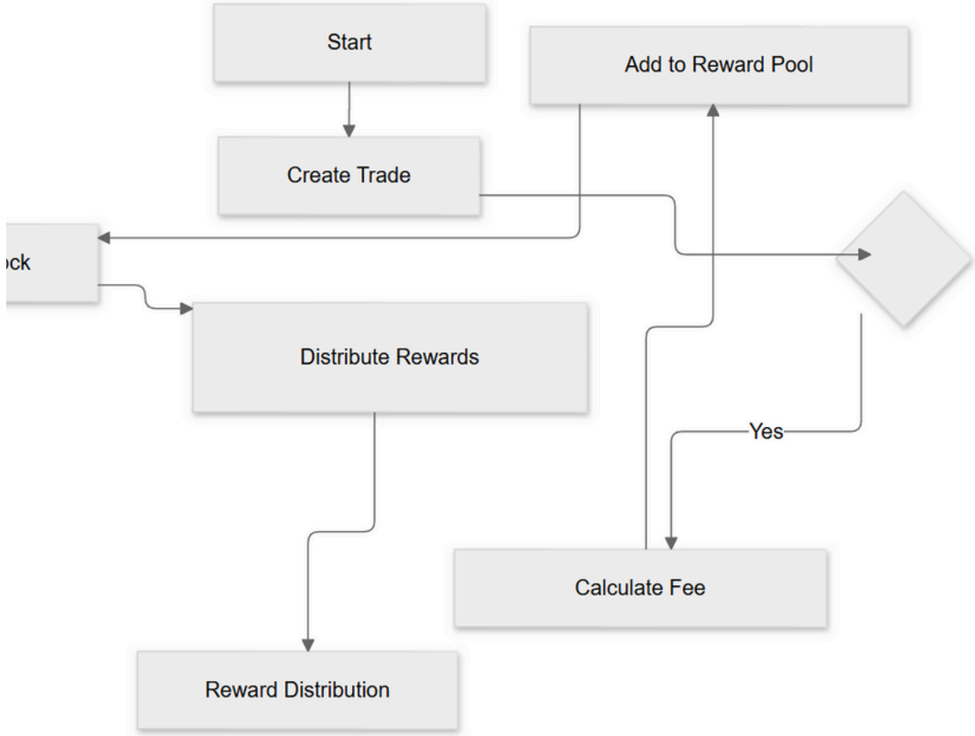


Portability Contracts and Economic Token Model



We use Ownable2Step to provide a secure ownership management and transfer mechanism for subsequent owner updates, ensuring that the management rights of the contract will not be easily abused.
Pausable allows the contract to suspend operations when necessary, thereby enhancing security and protecting users and contract assets. By combining these two contracts, a flexible and secure contract management system can be implemented.

EconomyLib is a library for managing trading economic models, including fee calculation, reward pool management, and trading state definition.

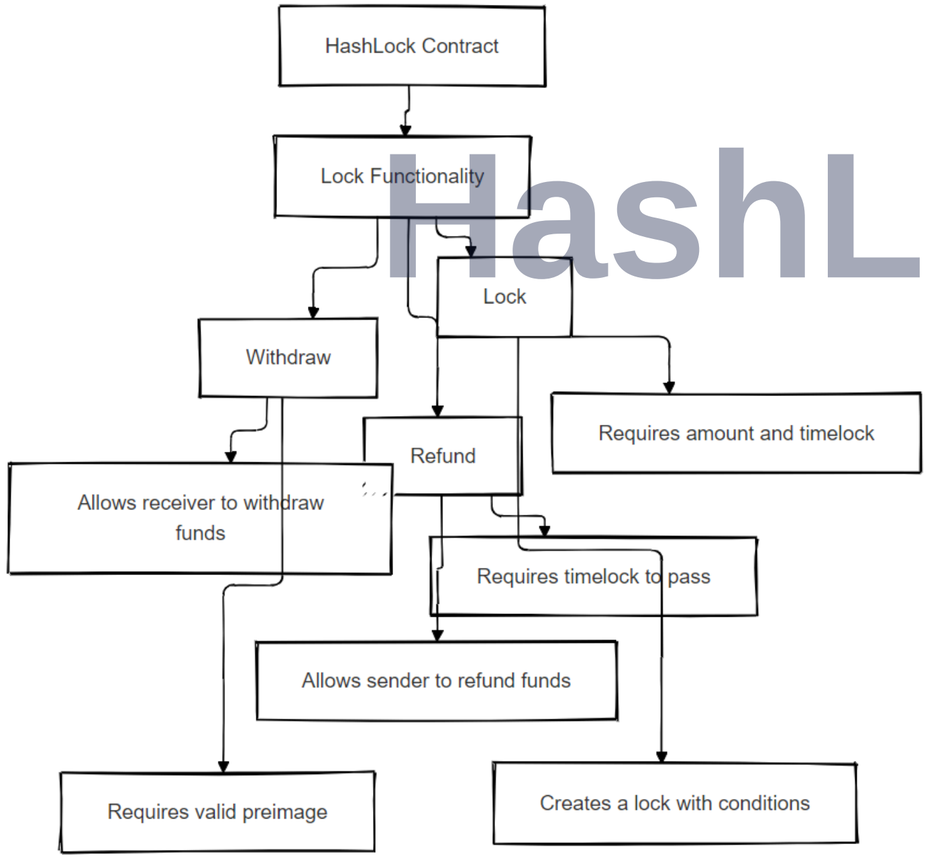


05

Privacy and Security Protection

```
struct Lock {
    uint256 amount;
    HomomorphicEncryptionLib.EncryptedData hashLock;
    bytes32 hashLock;
    uint256 timelock;
    address payable sender;
    address payable receiver;
    bool withdrawn;
    bool refunded;
    bytes32 preimage;
}
```

```
function lock(bytes32 _hashLock, uint256 _timelock, address payable _receiver) internal whenNotPaused returns (bytes32 lockId) {
    require(msg.value > 0, "Amount must be greater than 0");
    require(_timelock > block.timestamp, "Timelock must be in the future");

    lockId = keccak256(abi.encodePacked(msg.sender, _receiver, msg.value, _hashLock, _timelock));
    require(locks[lockId].sender == address(0), "Lock already exists");



    HomomorphicEncryptionLib.KeyPair memory key = HomomorphicEncryptionLib.generateKeyPair();
    HomomorphicEncryptionLib.EncryptedData memory encryptedHashLock = HomomorphicEncryptionLib.encrypt(_hashLock, key);

    locks[lockId] = Lock({
        amount: msg.value,
        hashLock: _hashLock,
        hashLock: encryptedHashLock,
        timelock: _timelock,
        sender: payable(msg.sender),
        receiver: _receiver,
        withdrawn: false,
        refunded: false,
        preimage: bytes32(0)
    });

    emit Locked(lockId, msg.sender, _receiver, msg.value, _hashLock, _timelock);
}
```



HashLock mechanism

Lock in funds:
The sender calls the lock function to lock the funds. This function receives hash locks (based on pre image generated hash values), time locks, and receiver addresses. The locked funds will be stored on the chain in a Lock structure.

Transaction identifier:
Each locked fund has a unique lock ID generated by hashing the relevant parameters. This ensures the uniqueness of each transaction.

Withdrawal of funds:
The receiver needs to call the withdraw function and provide a pre image. The contract will check whether the pre image matches the stored hash lock. The recipient can only successfully withdraw funds if they match and have not been withdrawn before.

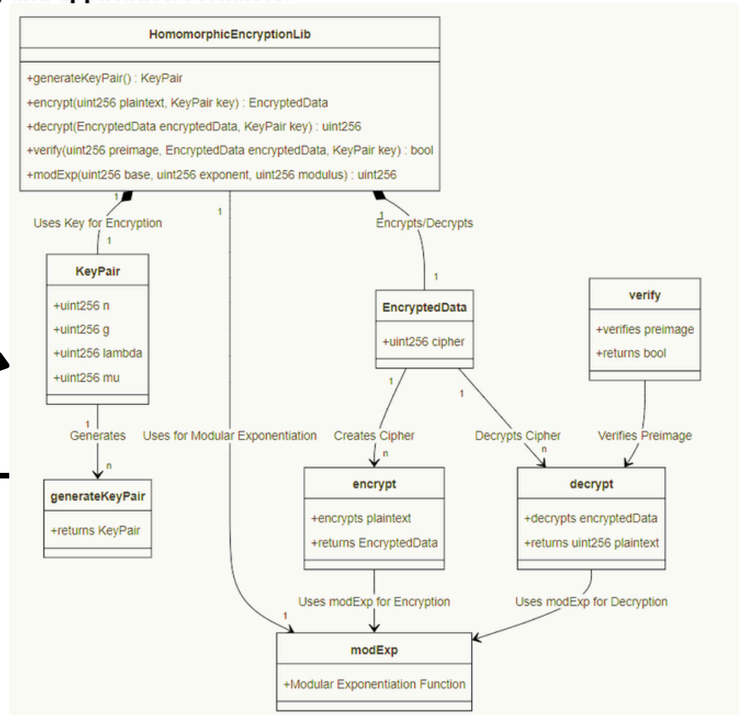
Refund mechanism:
If the time lock expires and the funds have not been withdrawn, the sender can retrieve the funds through the refund function. This provides protection for the sender to avoid losing funds due to incomplete transactions.

Event record:
The contract defines multiple events (such as Locked Withdrawn、Refunded） ， Each operation can be tracked on the blockchain. This enhances transparency and traceability.

Security:
This mechanism combines time lock and hash lock to ensure that funds are in a secure state before transactions are completed, preventing fraudulent behavior.

Homomorphic encryption is a special encryption technique that allows operations to be performed on encrypted data without decrypting it first. It makes data processing more secure and flexible while protecting data privacy. The following is a detailed introduction to homomorphic encryption, including its principles, implementation methods, advantages and disadvantages, and application scenarios.

Architecture Diagram



Key pair generation:
The generateKeyPair function generates a key pair, which includes:
Modulus n: The foundation of encryption algorithms.
Base g: A constant used for encryption and decryption.
Lambda: The ϕ value of n, used for decryption calculations.
μ (mu): Private key used to recover plaintext.

Encryption process:
In the encrypt function, plaintext data (such as a number) is encrypted. Using a random number r during the encryption process ensures the uniqueness of each encryption result, thereby enhancing security. The calculated encrypted data is stored in the EncryptdData structure.

Decryption process:
The decrypt function is responsible for converting encrypted data back to plaintext. During this process, modular exponentiation was used to achieve decryption.

Verification function:
The verify function decrypts encrypted data and compares it with the pre image to ensure that the provided pre image is correct and avoid erroneous fund withdrawals.

Modular power operation:
The modExp function implements modular exponentiation, which is one of the core operations in homomorphic encryption and is used for computation during encryption and decryption processes.

Combining HashLock mechanism
In your HashLock mechanism, homomorphic encryption can be used to encrypt hash locks. This combination provides additional security and privacy protection. For example:
Secure storage hash lock: By homomorphic encryption, the hash lock is stored to ensure that even if the contract is accessed, outsiders cannot obtain the original pre image information.
Unlocking mechanism: When the receiver requests to withdraw funds, the contract can use homomorphic decryption to verify the validity of the pre image without directly exposing the original data.

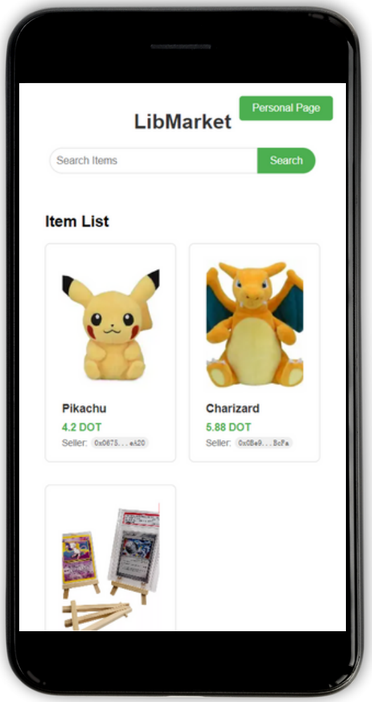
```solidity
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

library HomomorphicEncryptionLib {
    struct EncryptedData {
        uint256 cipher;
    }

    struct KeyPair {
        uint256 n;
        uint256 g;
        uint256 lambda;
        uint256 mu;
    }

    function generateKeyPair() internal pure returns (KeyPair memory) {
        uint256 n = 221;
        uint256 g = n + 1;
        uint256 lambda = 110;
        uint256 mu = 1;
        return KeyPair(n, g, lambda, mu);
    }

    function encrypt(uint256 plaintext, KeyPair memory key) internal view returns (EncryptedData memory) {
        require(plaintext > 0, "Plaintext must be positive");
        uint256 r = uint256(keccak256(abi.encodePacked(block.timestamp))) % key.n;
        uint256 cipher = (modExp(key.g, plaintext, key.n * key.n) * modExp(r, key.n, key.n * key.n)) % (key.n * key.n);
        return EncryptedData(cipher);
    }

    function decrypt(EncryptedData memory encryptedData, KeyPair memory key) internal view returns (uint256) {
        uint256 nSquared = key.n * key.n;
        uint256 u = modExp(encryptedData.cipher, key.lambda, nSquared); // u = c^λ mod n^2
        uint256 plaintext = (u - 1) / key.n * key.mu % key.n;
        return plaintext;
    }

    function verify(uint256 preimage, EncryptedData memory encryptedData, KeyPair memory key) internal view returns (bool) {
        uint256 decryptedValue = decrypt(encryptedData, key);
        return decryptedValue == preimage;
    }

    function modExp(uint256 base, uint256 exponent, uint256 modulus) internal pure returns (uint256 result) {
        result = 1;
        base = base % modulus;
        while (exponent > 0) {
            if (exponent % 2 == 1) {
                result = (result * base) % modulus;
            }
            exponent = exponent >> 1;
            base = (base * base) % modulus;
        }
    }
}
```

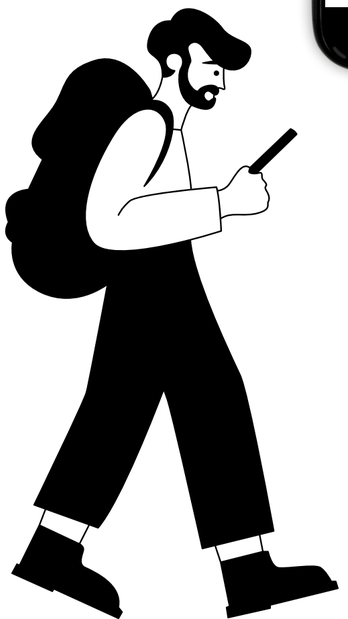
Code Block

06

Future plans

Mobile Application Development



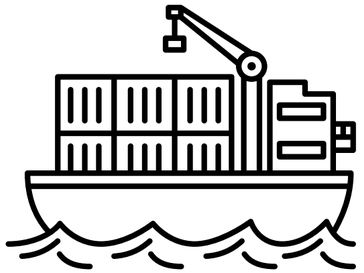


We will adopt a mobile-first design, ensuring the application is centered around mobile devices, optimizing the user interface and interactions to provide a high-quality experience.

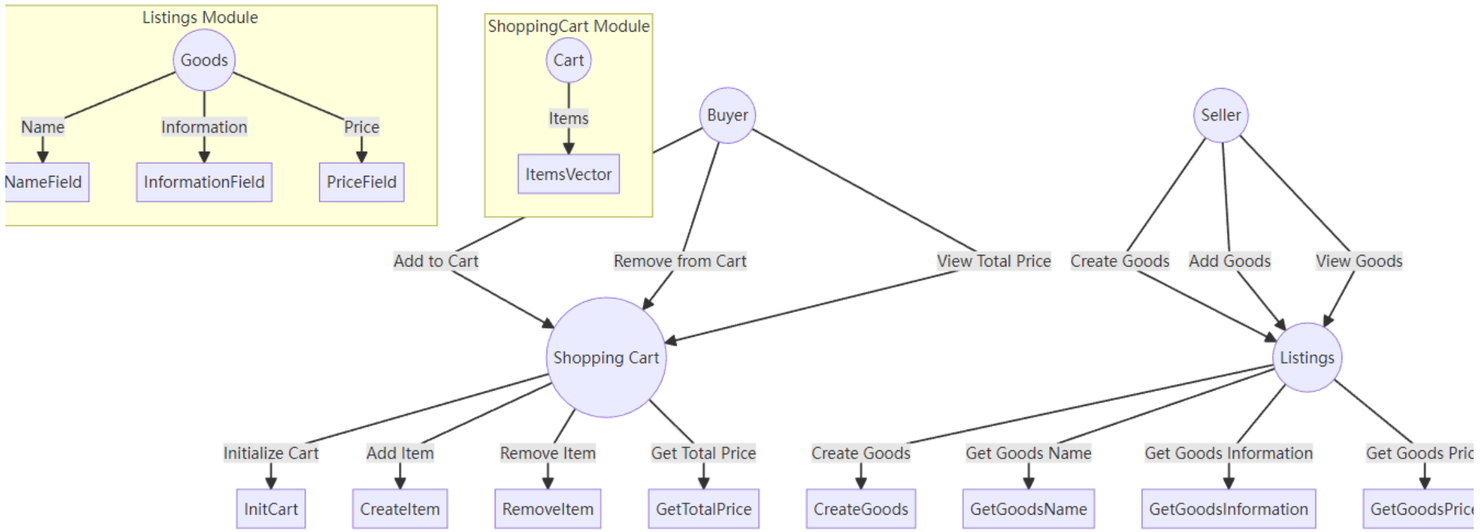
At the same time, we will implement real-time trading, price updates, and notifications through the mobile application, allowing users to stay informed about market dynamics at all times.

We will integrate biometric technologies (such as fingerprint or facial recognition) to ensure the security of user accounts and transactions.

Finally, we will provide global logistics to enhance delivery efficiency.

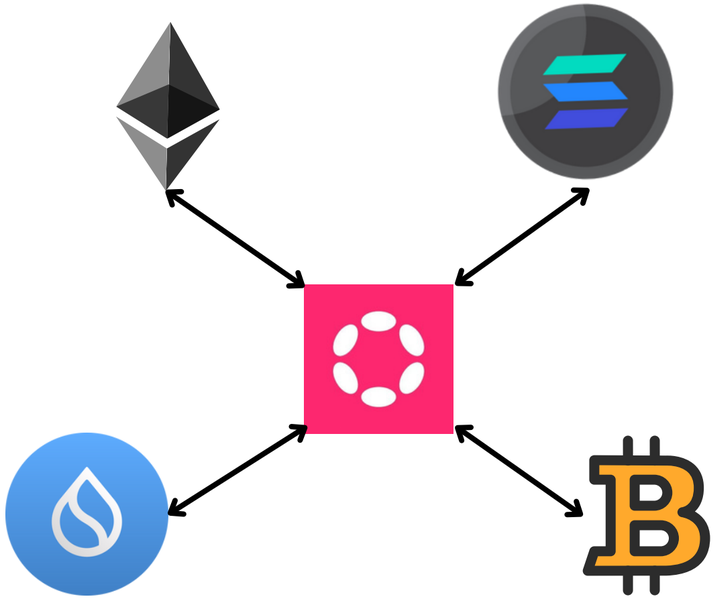


Multi-Ecosystem Interaction Expansion



In the future, we will also introduce multiple ecosystems to support trading with various currencies. At that time, you will be able to convert the cryptocurrencies you hold into the desired cryptocurrencies on our platform. This initiative will significantly expand our user base and enhance the user experience.

We have reserved interfaces so that in the future, we can connect the contract codes of other ecosystems through Polkadot. For example, sui can be used to implement shopping carts, lists and other functions. Other projects can call these interfaces, and contracts of other ecosystems can be used in the Polkadot ecosystem, expanding its coverage.



Project Vision

In this rapidly evolving digital age, traditional e-commerce models can no longer meet our growing needs, as the information requirements for registering on platforms lead to privacy breaches. We believe everyone should have a secure, transparent, and fair trading environment; this is not only about buying and selling goods but also about building trust and community. A decentralized e-commerce platform only requires a wallet address.

We are committed to breaking down the barriers of traditional commerce, allowing everyone to trade freely here. Many traditional platforms cannot serve global users due to geographical restrictions. For example, in China, many people find it very difficult to pay on well-known platforms like Amazon. Our platform dismantles these barriers, enabling anyone, regardless of location, to easily engage in global transactions, connecting buyers and sellers and promoting the globalization of commerce.

True commerce is not isolated transactions; it is about connections between people. We aim to create a community where sellers and buyers can communicate, collaborate, and jointly drive innovation and development in business. Whether you are a small creator or a consumer seeking personalized products, you will find your space on our platform.
On our future journey, we will continuously listen to the voices of the community and optimize the platform's functions and experiences. Our goal is not only to create a trading market but also to establish a vibrant ecosystem where every user can find value and a sense of belonging.

Let us reshape e-commerce together, creating a boundary-less trading environment and enjoying the convenience of global transactions.

07

About Us

Our Team

S7iter

PM&Web3sec,
Full stack dev

Azhan

CD&Web3sec,
backend dev

Ch1hiro

Backend Dev&
Security Researcher

LittleNewbie

Front Dev

UPON

Cryptography and
Market research

Decentralized Open Free Trade Platform