区块链拍卖系统

汇报人：刘伊蕾

目录
Direction

主要任务

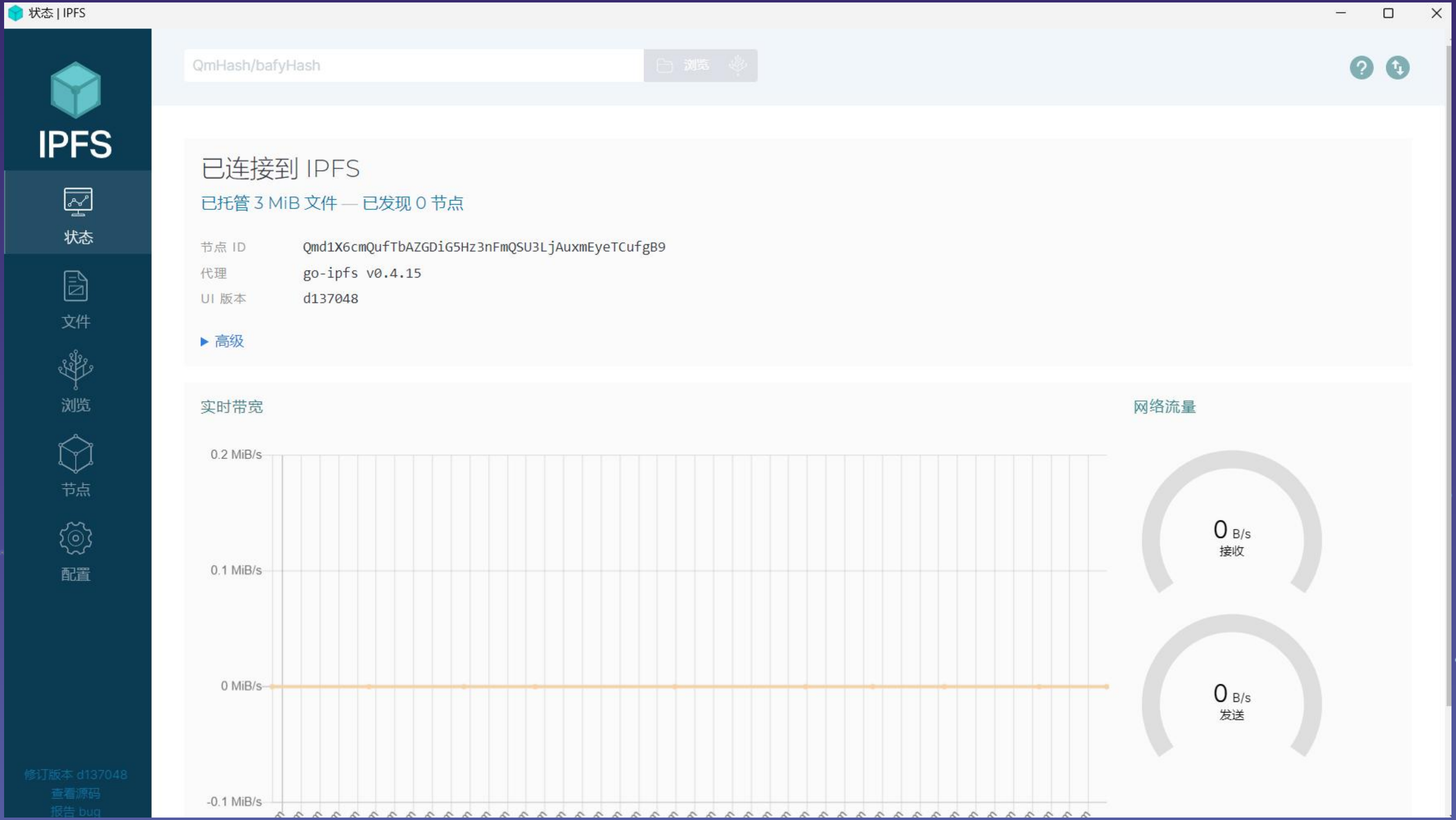
- 基于区块链平台实现商品拍卖功能

- 在区块链平台上实现商品展示与商品文件存储，具有将商品图像和商品描述(大文本)上传至IPFS的功能。
- 具有用户可根据类别、拍卖时间等过滤和浏览商品的功能。
- 实现维克里密封拍卖。

目录
Direction

IPFS Desktop

避免了中心节点失效，无审查和管控的完全去中心化的点到点传输网络



已连接到 IPFS

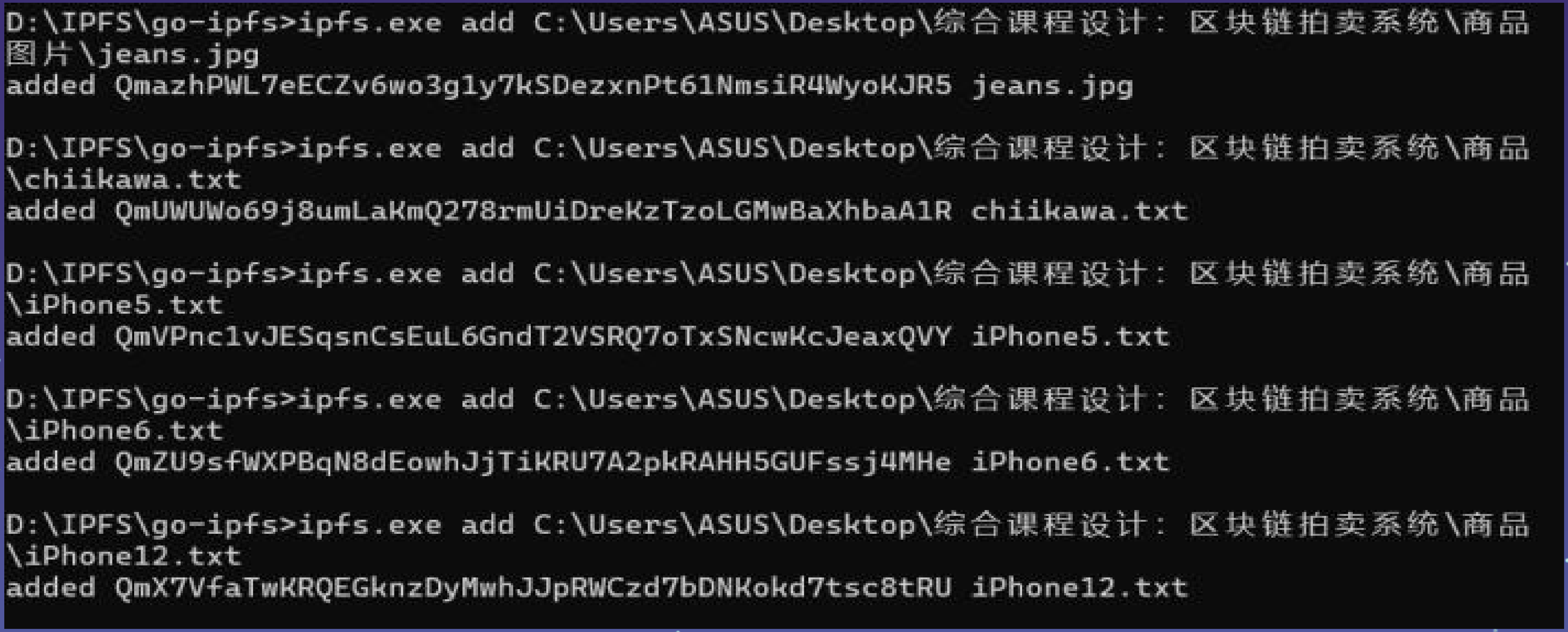
已托管 3 MiB 文件 — 已发现 0 节点

节点 ID	Qmd1X6cmQufTbAZGDiG5Hz3nFmQSU3LjAuxmEyeTCufgB9
代理	go-ipfs v0.4.15
UI 版本	d137048

▶ 高级

IPFS命令行

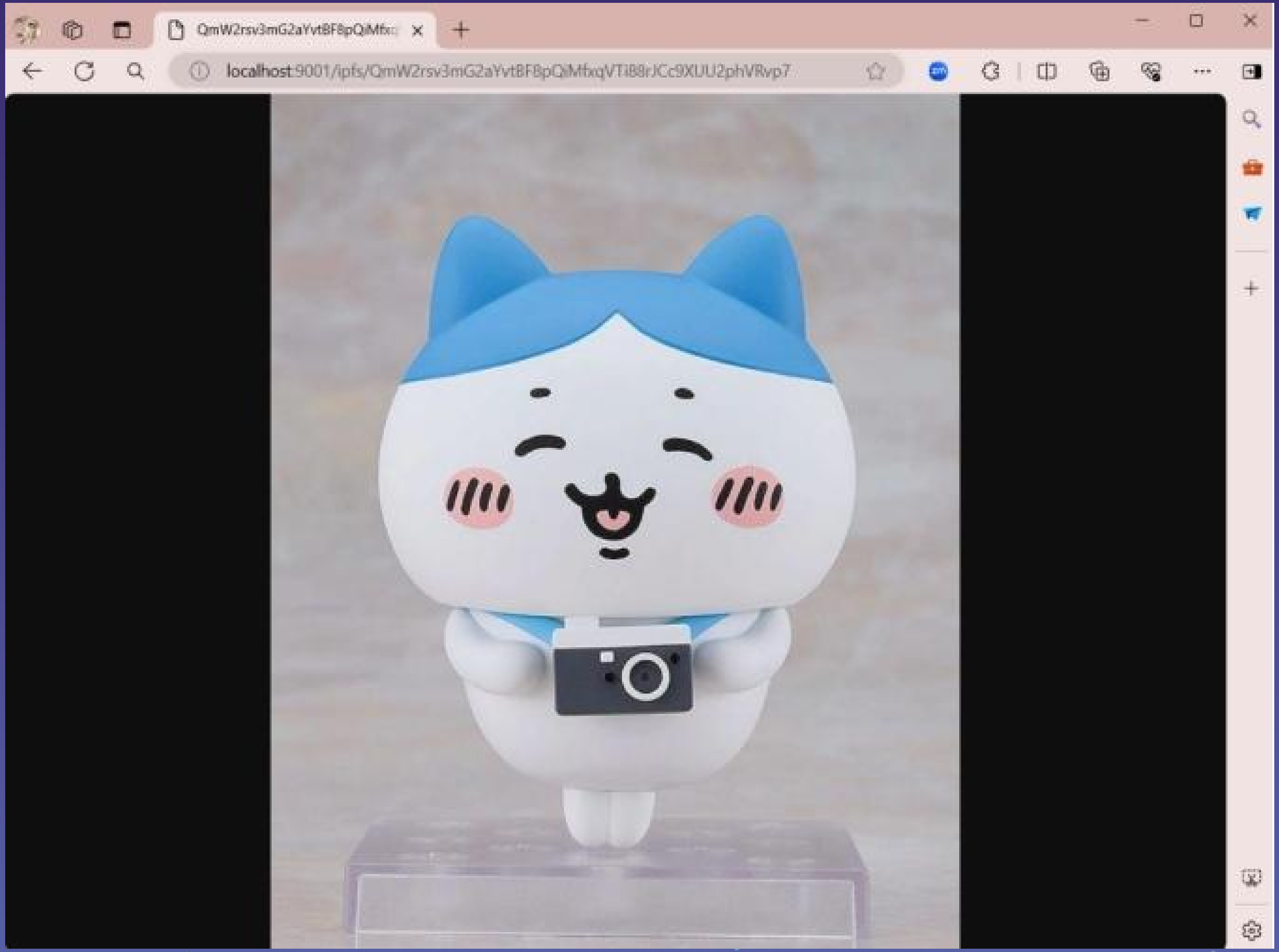
上传命令：ipfs add C:\Users\ASUS\Desktop\Recently\usagi.jpg
获得一串hash：QmdKEDNMPgZzZ1Bd3q6pf2gnEpyqRkEsP9dHLJCkXTxcCF



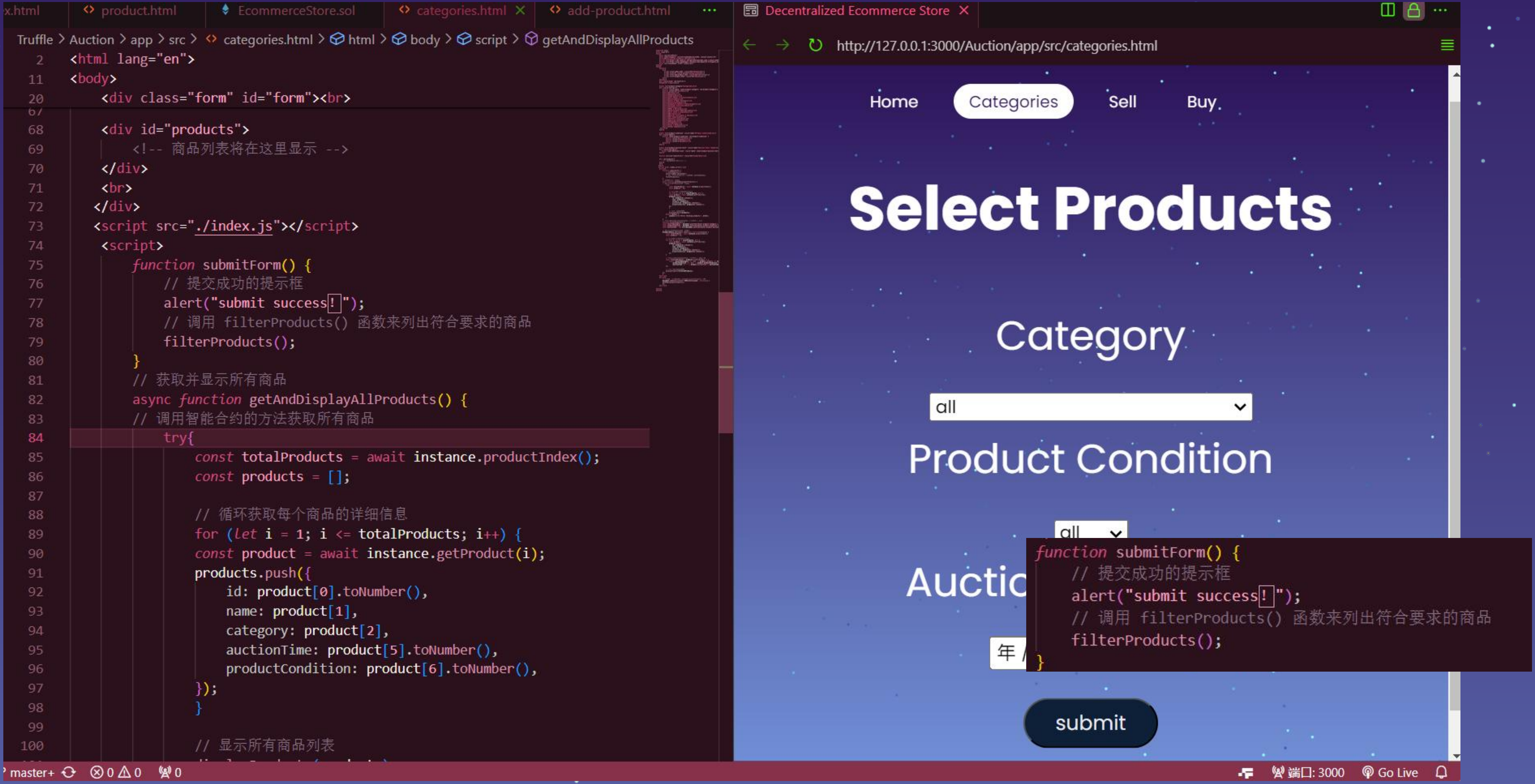
```
D:\IPFS\go-ipfs>ipfs.exe add C:\Users\ASUS\Desktop\综合课程设计：区块链拍卖系统\商品
图片\jeans.jpg
added QmazhPWL7eECZv6wo3gly7kSDezxnPt61NmsiR4WyoKJR5 jeans.jpg

D:\IPFS\go-ipfs>ipfs.exe add C:\Users\ASUS\Desktop\综合课程设计：区块链拍卖系统\商品
\chiikawa.txt
added QmUWUWo69j8umLaKmQ278rmUiDreKzTzoLGMwBaXhbaA1R chiikawa.txt

D:\IPFS\go-ipfs>ipfs.exe add C:\Users\ASUS\Desktop\综合课程设计：区块链拍卖系统\商品
\iPhone5.txt
added QmVPnclvJESqsnCsEuL6GndT2VSRQ7oTxSNcwKcJeaxQVY iPhone5.txt

D:\IPFS\go-ipfs>ipfs.exe add C:\Users\ASUS\Desktop\综合课程设计：区块链拍卖系统\商品
\iPhone6.txt
added QmZU9sfWXPBqN8dEowhJjTiKRU7A2pkRAHH5GUFssj4MHe iPhone6.txt

D:\IPFS\go-ipfs>ipfs.exe add C:\Users\ASUS\Desktop\综合课程设计：区块链拍卖系统\商品
\iPhone12.txt
added QmX7VfaTwKRQEGknzDyMwhJJpRWCzd7bDNKokd7tsc8tRU iPhone12.txt
```

IPFS命令行

- 将文件上传到IPFS成功后会自动获得一个hash值，只需要保存下这个哈希值就可以访问到文件了
  访问：http://localhost:9001/ipfs/QmW2rsv3mG2aYvtBF8pQiMfxqVTi88rJCc9XUU2phVRvp7


QmW2rsv3mG2aYvtBF8pQiMfx
localhost:9001/ipfs/QmW2rsv3mG2aYvtBF8pQiMfxqVTi88rJCc9XUU2phVRvp7

过滤和浏览商品

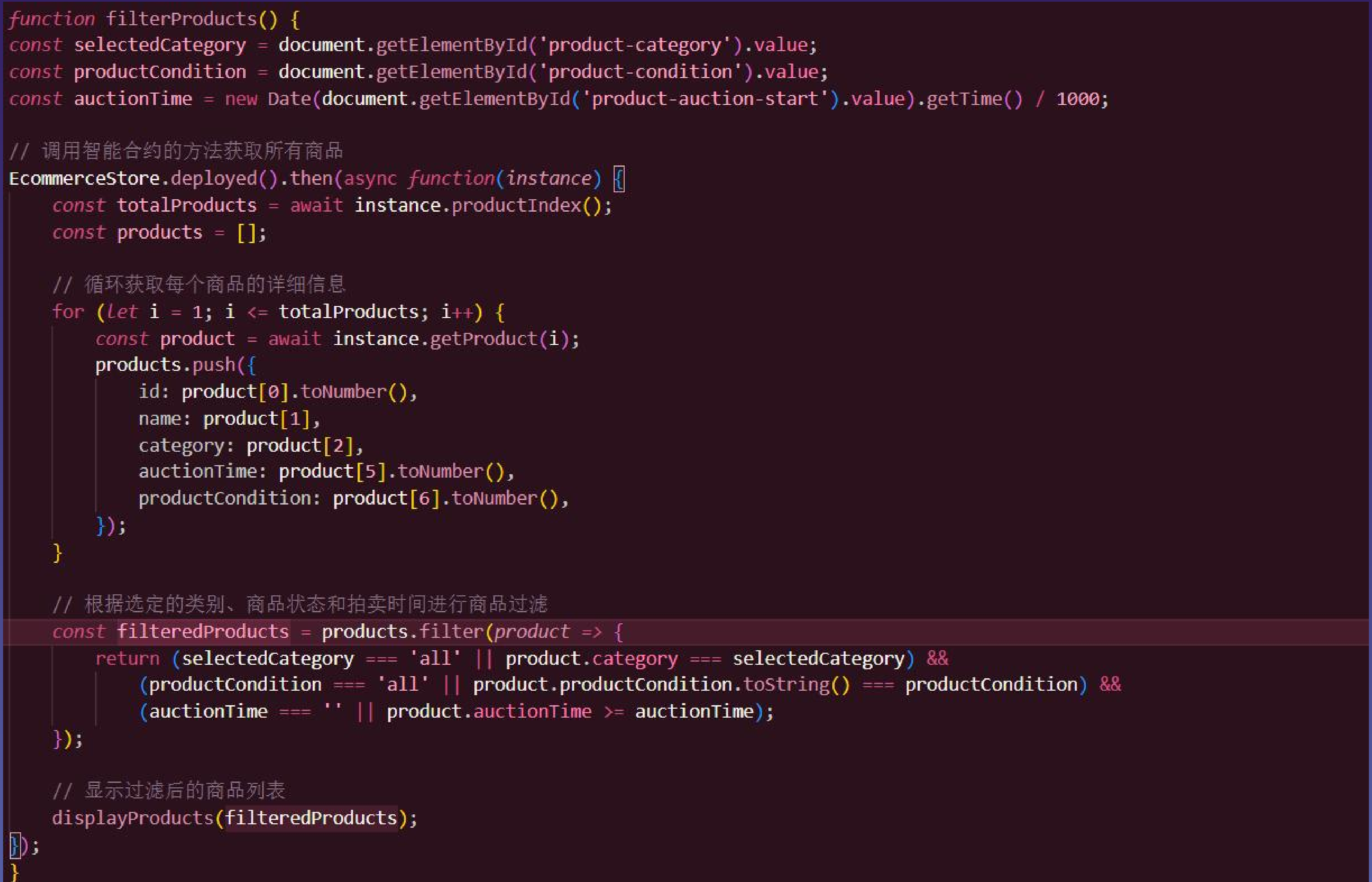


```html
<html lang="en">
<body>
    <div class="form" id="form"><br>

    <div id="products">
        <!-- 商品列表将在这里显示 -->
    </div>
    <br>
   </div>
   <script src="./index.js"></script>
    <script>
        function submitForm() {
            // 提交成功的提示框
            alert("submit success!");
            // 调用 filterProducts() 函数来列出符合要求的商品
            filterProducts();
        }
        // 获取并显示所有商品
        async function getAndDisplayAllProducts() {
        // 调用智能合约的方法获取所有商品
            try{
                const totalProducts = await instance.productIndex();
                const products = [];

                // 循环获取每个商品的详细信息
                for (let i = 1; i <= totalProducts; i++) {
                const product = await instance.getProduct(i);
                products.push({
                    id: product[0].toNumber(),
                    name: product[1],
                    category: product[2],
                    auctionTime: product[5].toNumber(),
                    productCondition: product[6].toNumber(),
                });
                }

                // 显示所有商品列表
```

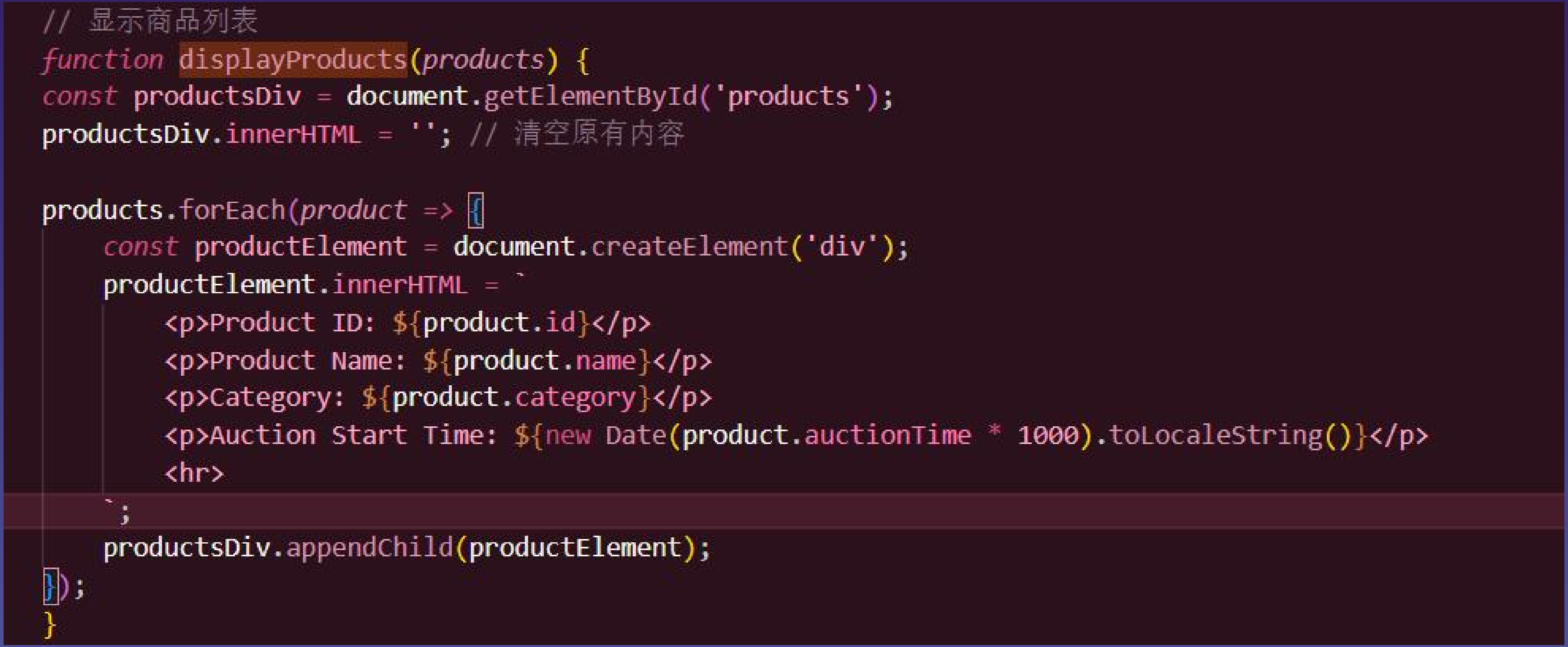
```javascript
function submitForm() {
    // 提交成功的提示框
    alert("submit success!");
    // 调用 filterProducts() 函数来列出符合要求的商品
    filterProducts();
}
```

x.html
product.html
EcommerceStore.sol
categories.html
add-product.html
Truffle > Auction > app > src > categories.html > html > body > script > getAndDisplayAllProducts
Decentralized Ecommerce Store
http://127.0.0.1:3000/Auction/app/src/categories.html
Home
Categories
Sell
Buy
Select Products
Category
all
Product Condition
all
Auctic
年 /
submit
master+
0 0
0
端口: 3000
Go Live

功能实现



```
function filterProducts() {
const selectedCategory = document.getElementById('product-category').value;
const productCondition = document.getElementById('product-condition').value;
const auctionTime = new Date(document.getElementById('product-auction-start').value).getTime() / 1000;

// 调用智能合约的方法获取所有商品
EcommerceStore.deployed().then(async function(instance) {
    const totalProducts = await instance.productIndex();
    const products = [];

    // 循环获取每个商品的详细信息
    for (let i = 1; i <= totalProducts; i++) {
        const product = await instance.getProduct(i);
        products.push({
            id: product[0].toNumber(),
            name: product[1],
            category: product[2],
            auctionTime: product[5].toNumber(),
            productCondition: product[6].toNumber(),
        });
    }

    // 根据选定的类别、商品状态和拍卖时间进行商品过滤
    const filteredProducts = products.filter(product => {
        return (selectedCategory === 'all' || product.category === selectedCategory) &&
            (productCondition === 'all' || product.productCondition.toString() === productCondition) &&
            (auctionTime === '' || product.auctionTime >= auctionTime);
    });

    // 显示过滤后的商品列表
    displayProducts(filteredProducts);
});
}
```



```
// 显示商品列表
function displayProducts(products) {
const productsDiv = document.getElementById('products');
productsDiv.innerHTML = ''; // 清空原有内容

products.forEach(product => {
    const productElement = document.createElement('div');
    productElement.innerHTML = `
        <p>Product ID: ${product.id}</p>
        <p>Product Name: ${product.name}</p>
        <p>Category: ${product.category}</p>
        <p>Auction Start Time: ${new Date(product.auctionTime * 1000).toLocaleString()}</p>
        <hr>
    `;
    productsDiv.appendChild(productElement);
});
}
```

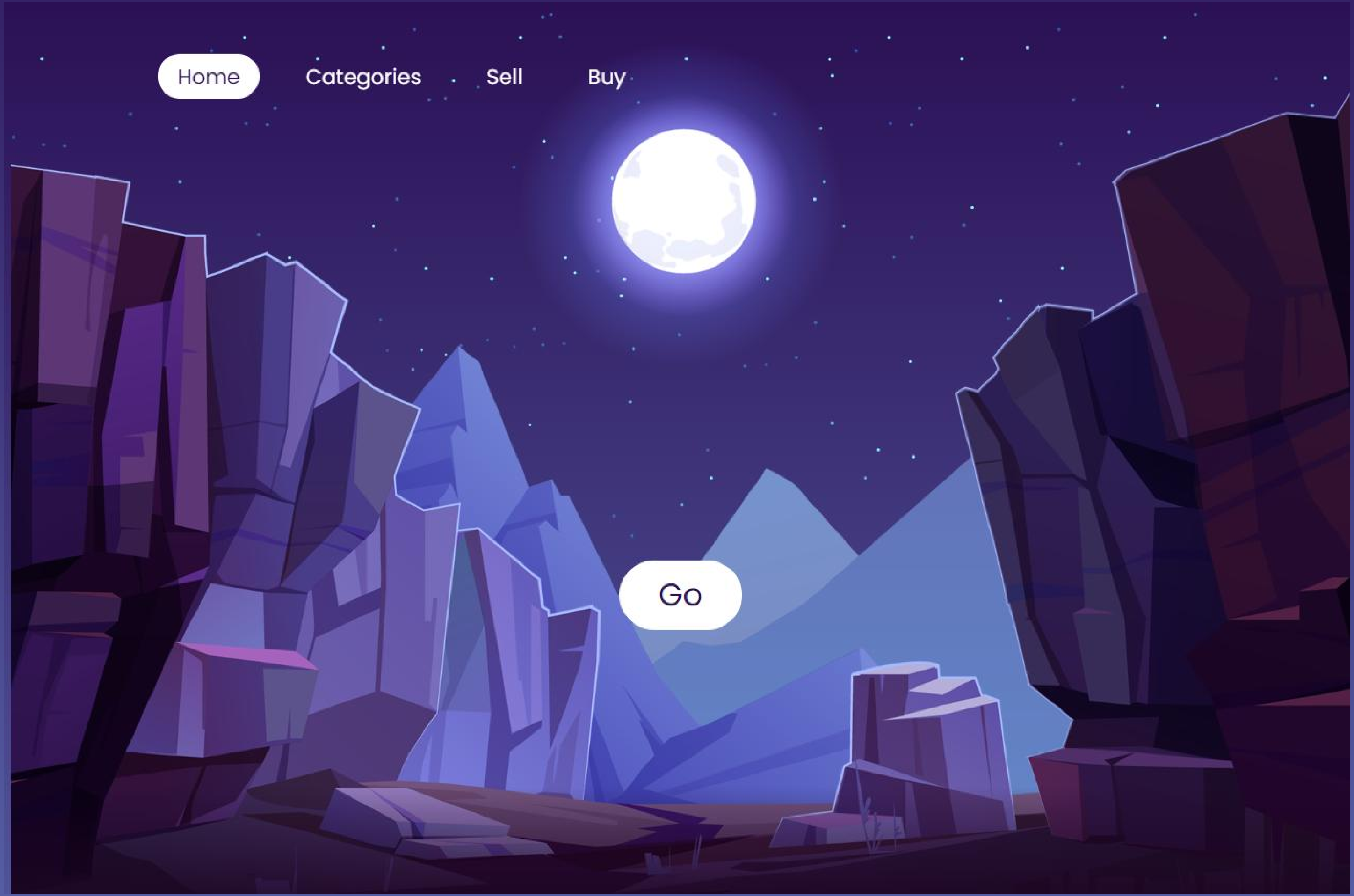
维克里密封拍卖

- 投标者在不知道其他人标价的情况下出价，且出价时必须同时支付ETH
- 标价最高者得标，且只需支付第二高的标价
- 所有输掉竞价的人将会收回各自出价的 ETH（扣除一些手续费）

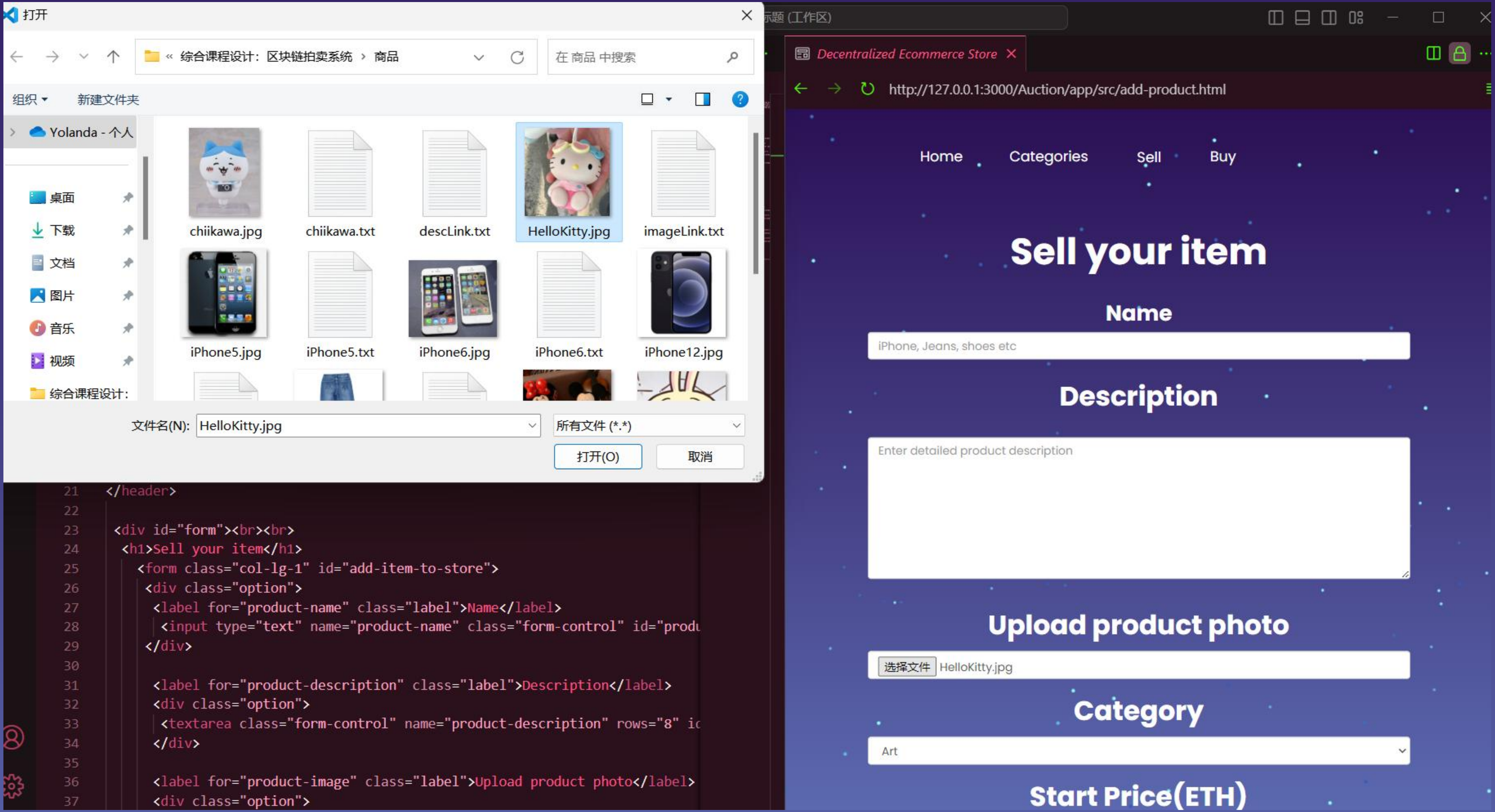
拓展功能：

美化html，提升用户体验

拍卖系统主界面



用户添加商品界面



打开

« 综合课程设计：区块链拍卖系统 › 商品

在 商品 中搜索

组织 ▾ 新建文件夹

Yolanda - 个人

桌面
下载
文档
图片
音乐
视频
综合课程设计：

chiikawa.jpg
chiikawa.txt
descLink.txt
HelloKitty.jpg
imageLink.txt
iPhone5.jpg
iPhone5.txt
iPhone6.jpg
iPhone6.txt
iPhone12.jpg

文件名(N): HelloKitty.jpg
所有文件 (*.*)
打开(O) 取消

```
</header>

 <div id="form"><br><br>
  <h1>Sell your item</h1>
    <form class="col-lg-1" id="add-item-to-store">
     <div class="option">
      <label for="product-name" class="label">Name</label>
       <input type="text" name="product-name" class="form-control" id="produ
     </div>

      <label for="product-description" class="label">Description</label>
      <div class="option">
       <textarea class="form-control" name="product-description" rows="8" ic
      </div>

      <label for="product-image" class="label">Upload product photo</label>
      <div class="option">
```

Decentralized Ecommerce Store

http://127.0.0.1:3000/Auction/app/src/add-product.html

Home Categories Sell Buy

Sell your item

Name

iPhone, Jeans, shoes etc

Description

Enter detailed product description

Upload product photo

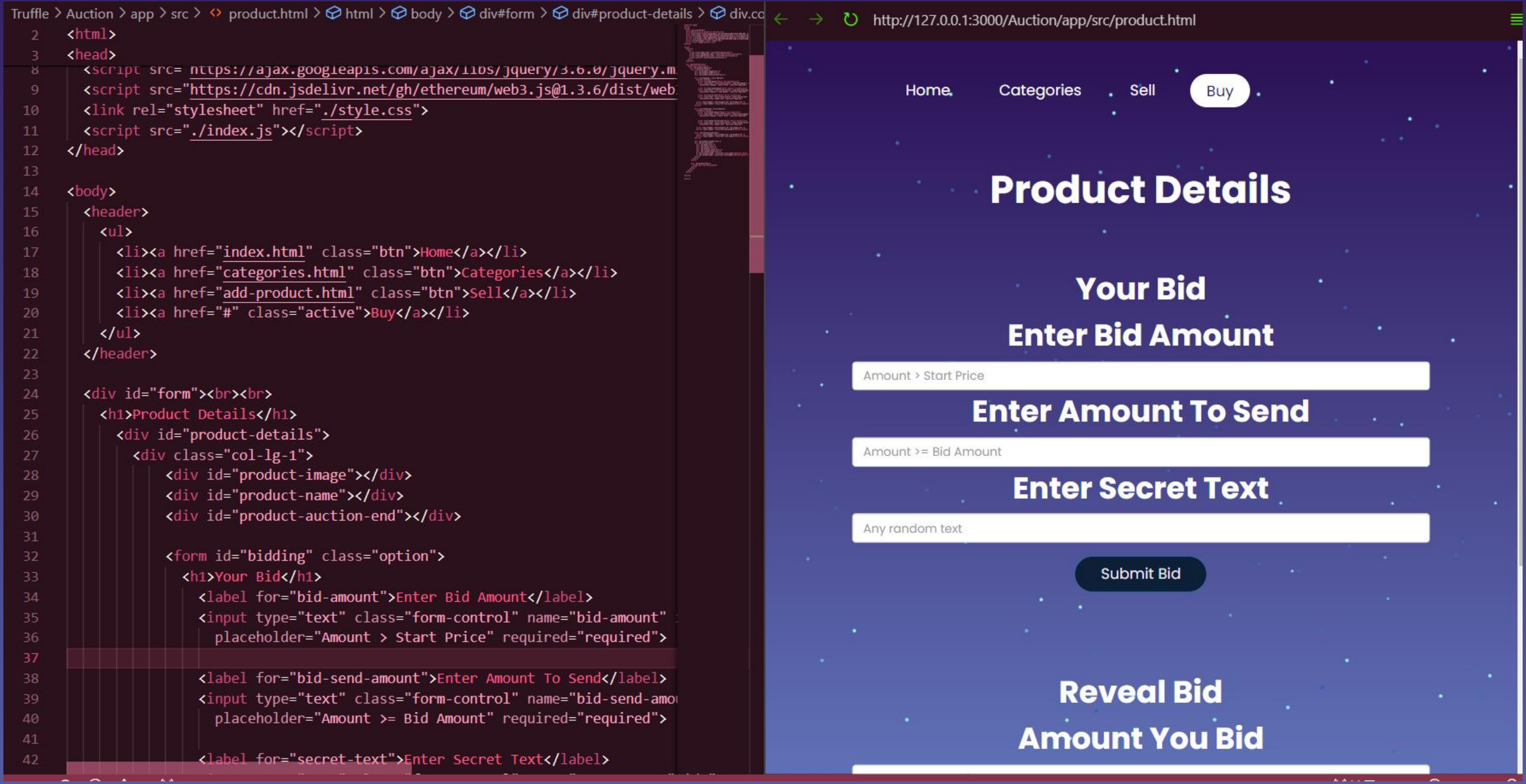
选择文件 HelloKitty.jpg

Category

Art

Start Price(ETH)

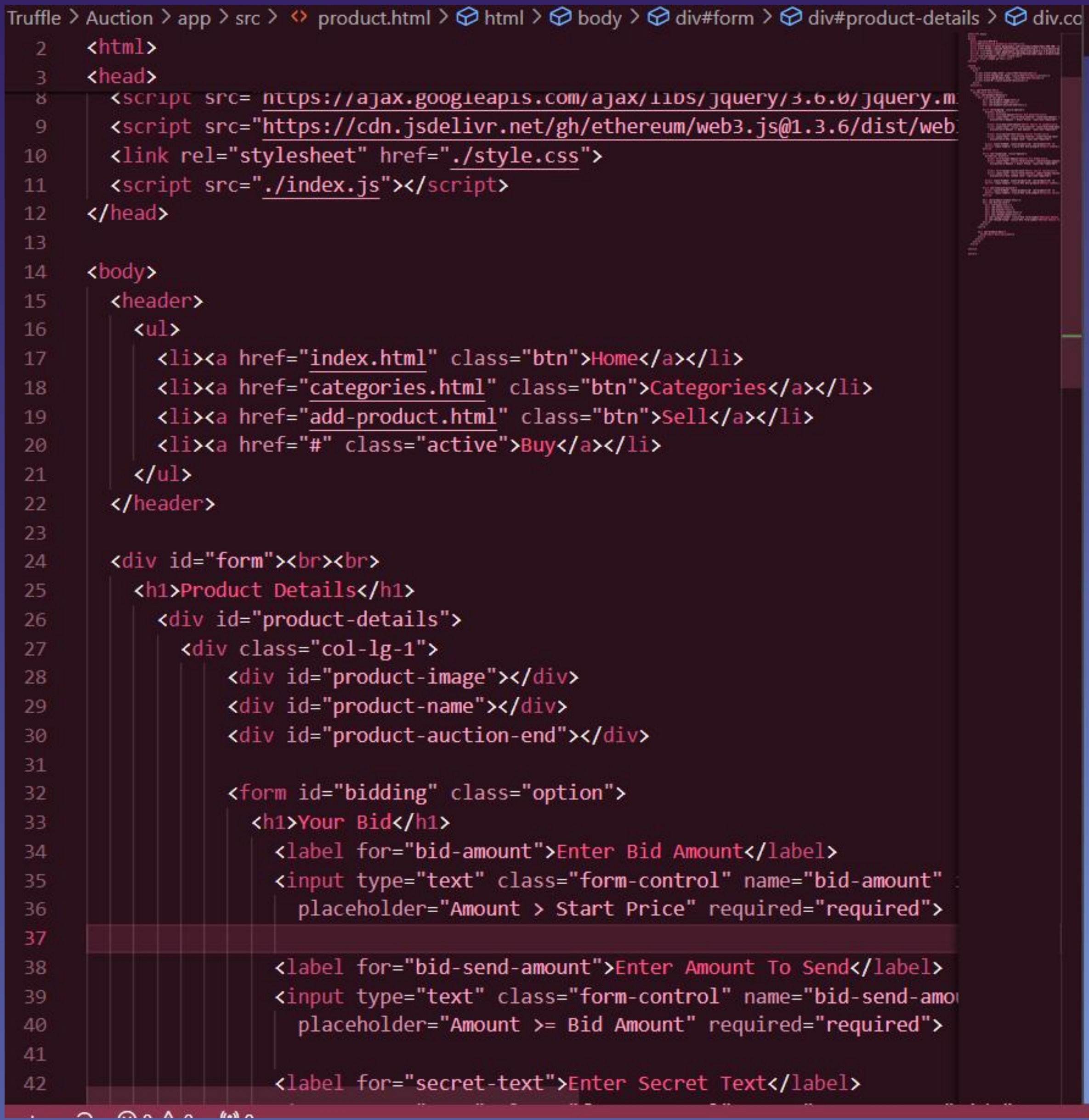
出价&揭示出价界面


Truffle > Auction > app > src > product.html > html > body > div#form > div#product-details > div.co

```html
<html>
<head>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.m
  <script src="https://cdn.jsdelivr.net/gh/ethereum/web3.js@1.3.6/dist/web
  <link rel="stylesheet" href="./style.css">
  <script src="./index.js"></script>
</head>

<body>
  <header>
    <ul>
      <li><a href="index.html" class="btn">Home</a></li>
      <li><a href="categories.html" class="btn">Categories</a></li>
      <li><a href="add-product.html" class="btn">Sell</a></li>
      <li><a href="#" class="active">Buy</a></li>
    </ul>
  </header>

  <div id="form"><br><br>
    <h1>Product Details</h1>
      <div id="product-details">
        <div class="col-lg-1">
            <div id="product-image"></div>
            <div id="product-name"></div>
            <div id="product-auction-end"></div>

            <form id="bidding" class="option">
              <h1>Your Bid</h1>
                <label for="bid-amount">Enter Bid Amount</label>
                <input type="text" class="form-control" name="bid-amount"
                  placeholder="Amount > Start Price" required="required">

                <label for="bid-send-amount">Enter Amount To Send</label>
                <input type="text" class="form-control" name="bid-send-amo
                  placeholder="Amount >= Bid Amount" required="required">

                <label for="secret-text">Enter Secret Text</label>
```

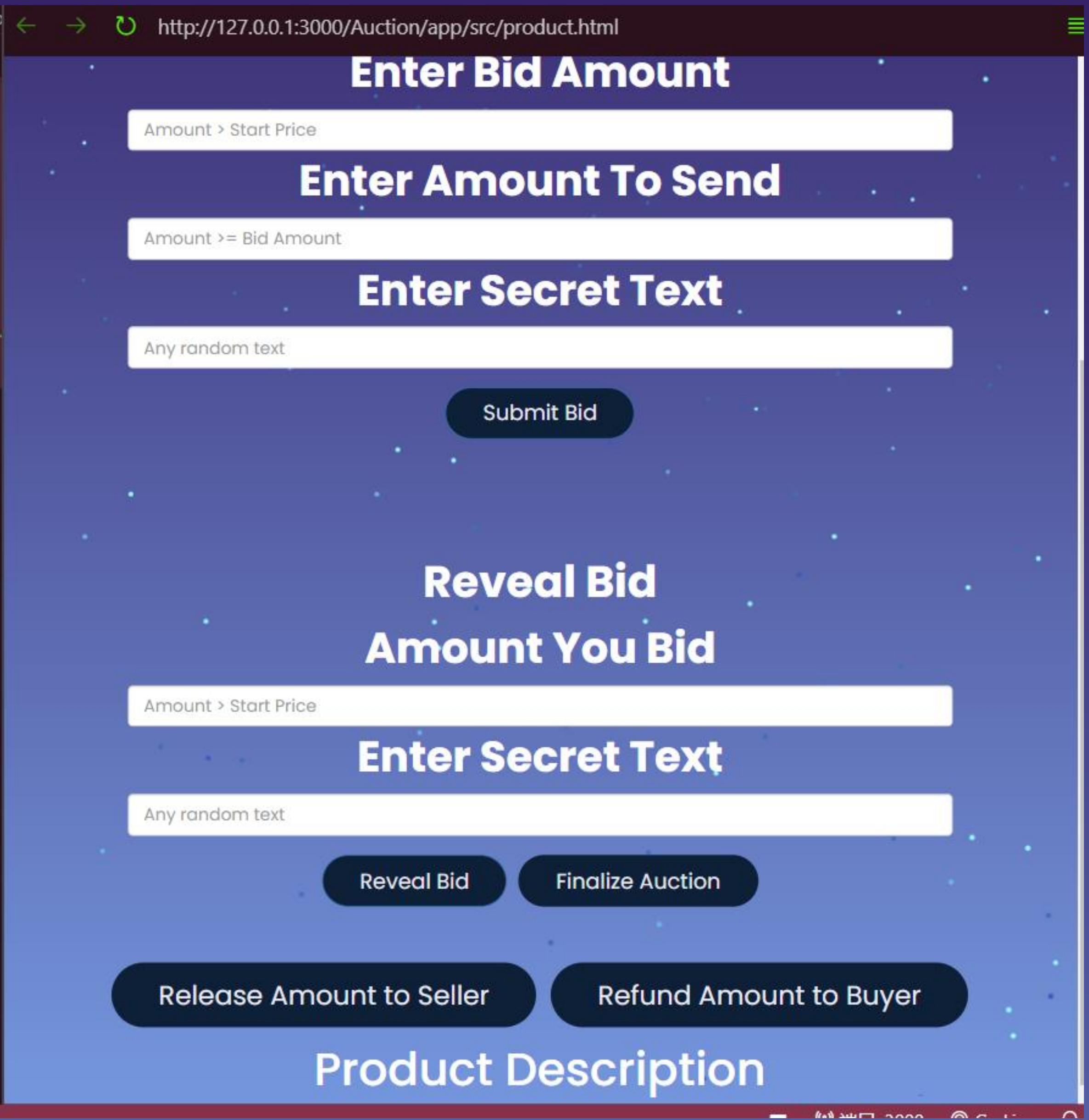
http://127.0.0.1:3000/Auction/app/src/product.html

出价&揭示出价界面



Truffle > Auction > app > src > product.html > html > body > div#form > div#product-details > div.co

```html
<html>
<head>
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.m
    <script src="https://cdn.jsdelivr.net/gh/ethereum/web3.js@1.3.6/dist/web
    <link rel="stylesheet" href="./style.css">
    <script src="./index.js"></script>
</head>

<body>
    <header>
        <ul>
            <li><a href="index.html" class="btn">Home</a></li>
            <li><a href="categories.html" class="btn">Categories</a></li>
            <li><a href="add-product.html" class="btn">Sell</a></li>
            <li><a href="#" class="active">Buy</a></li>
        </ul>
    </header>

    <div id="form"><br><br>
        <h1>Product Details</h1>
            <div id="product-details">
                <div class="col-lg-1">
                    <div id="product-image"></div>
                    <div id="product-name"></div>
                    <div id="product-auction-end"></div>

                    <form id="bidding" class="option">
                        <h1>Your Bid</h1>
                            <label for="bid-amount">Enter Bid Amount</label>
                            <input type="text" class="form-control" name="bid-amount"
                              placeholder="Amount > Start Price" required="required">

                            <label for="bid-send-amount">Enter Amount To Send</label>
                            <input type="text" class="form-control" name="bid-send-amo
                              placeholder="Amount >= Bid Amount" required="required">

                            <label for="secret-text">Enter Secret Text</label>
```



http://127.0.0.1:3000/Auction/app/src/product.html

Enter Bid Amount

Amount > Start Price

Enter Amount To Send

Amount >= Bid Amount

Enter Secret Text

Any random text

Submit Bid

Reveal Bid

Amount You Bid

Amount > Start Price

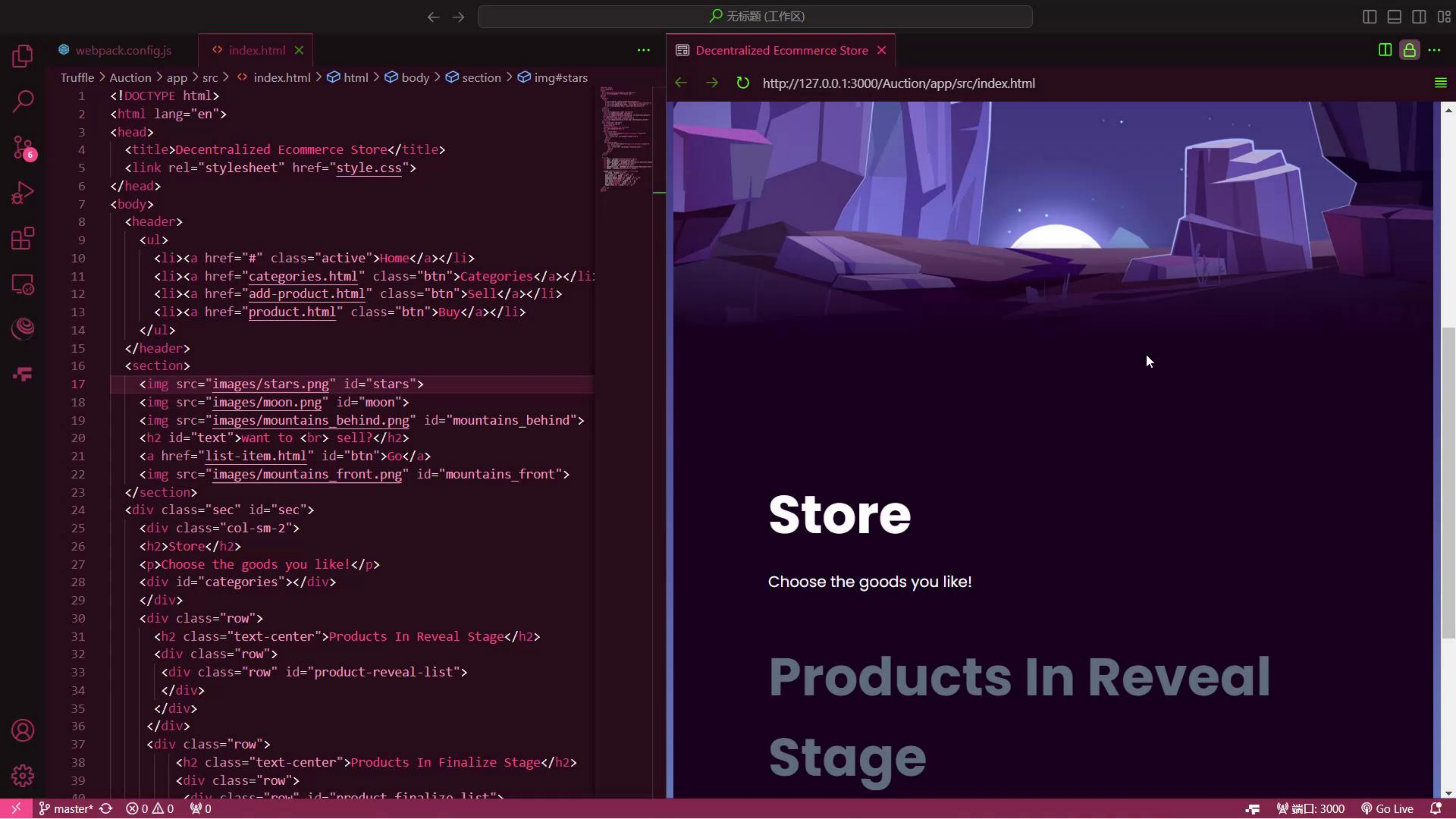
Enter Secret Text

Any random text

Reveal Bid | Finalize Auction

Release Amount to Seller | Refund Amount to Buyer

Product Description


无标题 (工作区)
webpack.config.js
index.html
Decentralized Ecommerce Store
Truffle > Auction > app > src > index.html > html > body > section > img#stars
http://127.0.0.1:3000/Auction/app/src/index.html
master*
0 0
0
端口: 3000
Go Live

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Decentralized Ecommerce Store</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <header>
    <ul>
      <li><a href="#" class="active">Home</a></li>
      <li><a href="categories.html" class="btn">Categories</a></li
      <li><a href="add-product.html" class="btn">Sell</a></li>
      <li><a href="product.html" class="btn">Buy</a></li>
    </ul>
  </header>
  <section>
    <img src="images/stars.png" id="stars">
    <img src="images/moon.png" id="moon">
    <img src="images/mountains_behind.png" id="mountains_behind">
    <h2 id="text">want to <br> sell?</h2>
    <a href="list-item.html" id="btn">Go</a>
    <img src="images/mountains_front.png" id="mountains_front">
  </section>
  <div class="sec" id="sec">
    <div class="col-sm-2">
    <h2>Store</h2>
    <p>Choose the goods you like!</p>
    <div id="categories"></div>
    </div>
    <div class="row">
      <h2 class="text-center">Products In Reveal Stage</h2>
      <div class="row">
       <div class="row" id="product-reveal-list">
       </div>
      </div>
     </div>
     <div class="row">
        <h2 class="text-center">Products In Finalize Stage</h2>
        <div class="row">
          <div class="row" id="product-finalize-list">
```

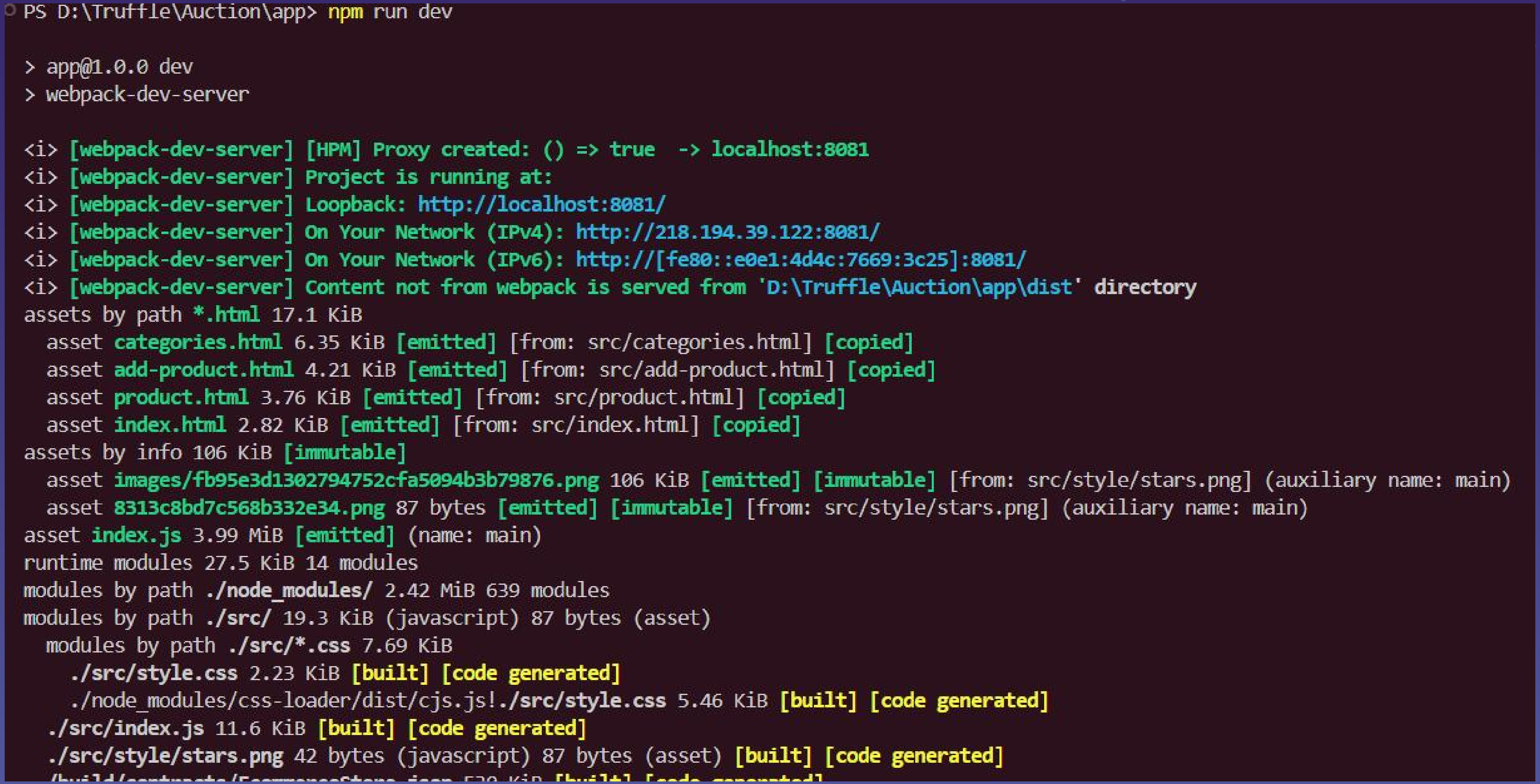
Store

Choose the goods you like!

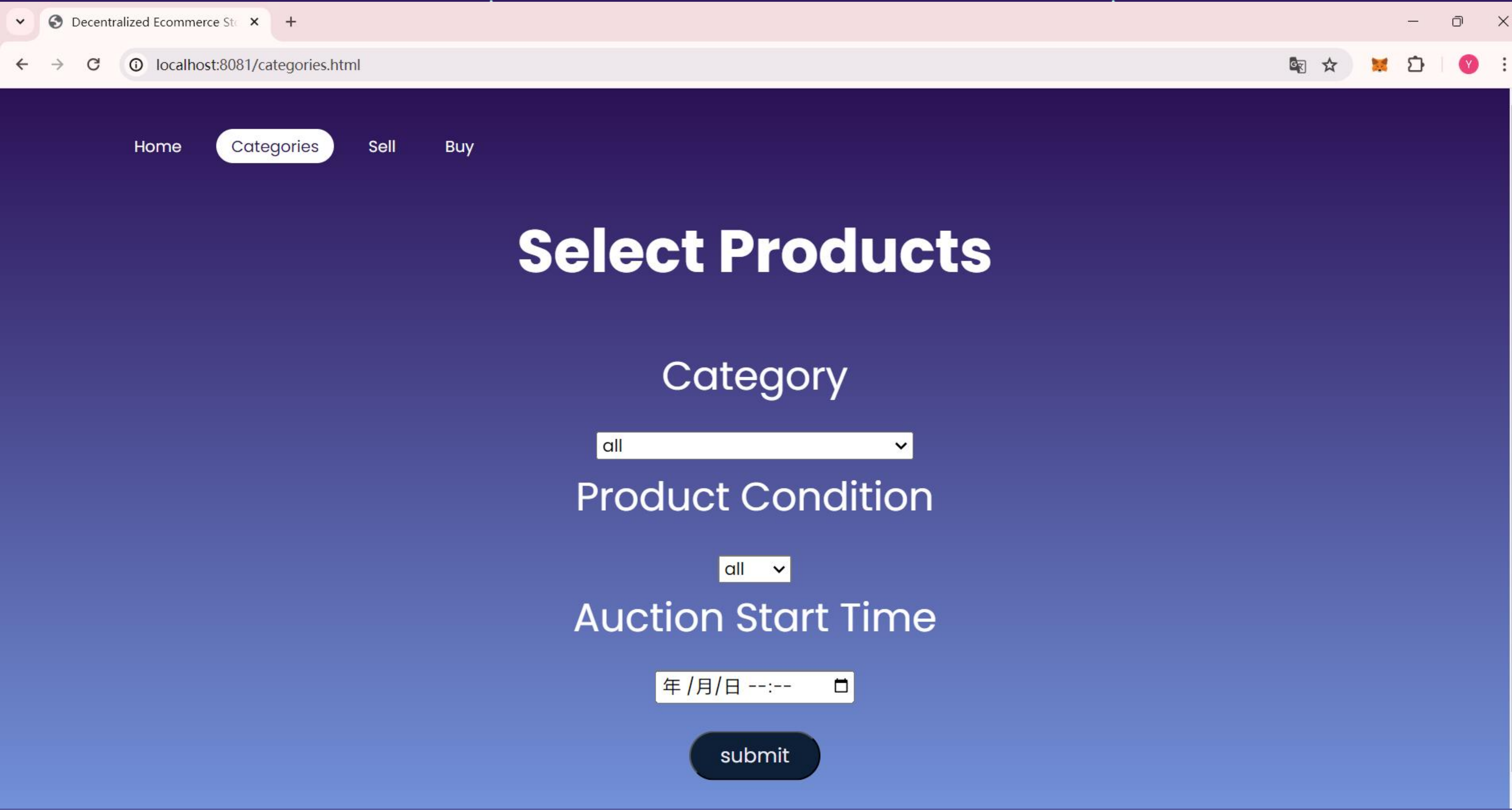
Products In Reveal Stage

目录
Direction

环境启动



```
PS D:\Truffle\Auction\app> npm run dev

> app@1.0.0 dev
> webpack-dev-server

<i> [webpack-dev-server] [HPM] Proxy created: () => true  -> localhost:8081
<i> [webpack-dev-server] Project is running at:
<i> [webpack-dev-server] Loopback: http://localhost:8081/
<i> [webpack-dev-server] On Your Network (IPv4): http://218.194.39.122:8081/
<i> [webpack-dev-server] On Your Network (IPv6): http://[fe80::e0e1:4d4c:7669:3c25]:8081/
<i> [webpack-dev-server] Content not from webpack is served from 'D:\Truffle\Auction\app\dist' directory
assets by path *.html 17.1 KiB
  asset categories.html 6.35 KiB [emitted] [from: src/categories.html] [copied]
  asset add-product.html 4.21 KiB [emitted] [from: src/add-product.html] [copied]
  asset product.html 3.76 KiB [emitted] [from: src/product.html] [copied]
  asset index.html 2.82 KiB [emitted] [from: src/index.html] [copied]
assets by info 106 KiB [immutable]
  asset images/fb95e3d1302794752cfa5094b3b79876.png 106 KiB [emitted] [immutable] [from: src/style/stars.png] (auxiliary name: main)
  asset 8313c8bd7c568b332e34.png 87 bytes [emitted] [immutable] [from: src/style/stars.png] (auxiliary name: main)
asset index.js 3.99 MiB [emitted] (name: main)
runtime modules 27.5 KiB 14 modules
modules by path ./node_modules/ 2.42 MiB 639 modules
modules by path ./src/ 19.3 KiB (javascript) 87 bytes (asset)
  modules by path ./src/*.css 7.69 KiB
    ./src/style.css 2.23 KiB [built] [code generated]
    ./node_modules/css-loader/dist/cjs.js!./src/style.css 5.46 KiB [built] [code generated]
  ./src/index.js 11.6 KiB [built] [code generated]
  ./src/style/stars.png 42 bytes (javascript) 87 bytes (asset) [built] [code generated]
  ./build/contracts/EcommerceStore.json 530 KiB [built] [code generated]
```


Decentralized Ecommerce St

localhost:8081/categories.html

Home · Categories · Sell · Buy

Select Products

Category

all

Product Condition

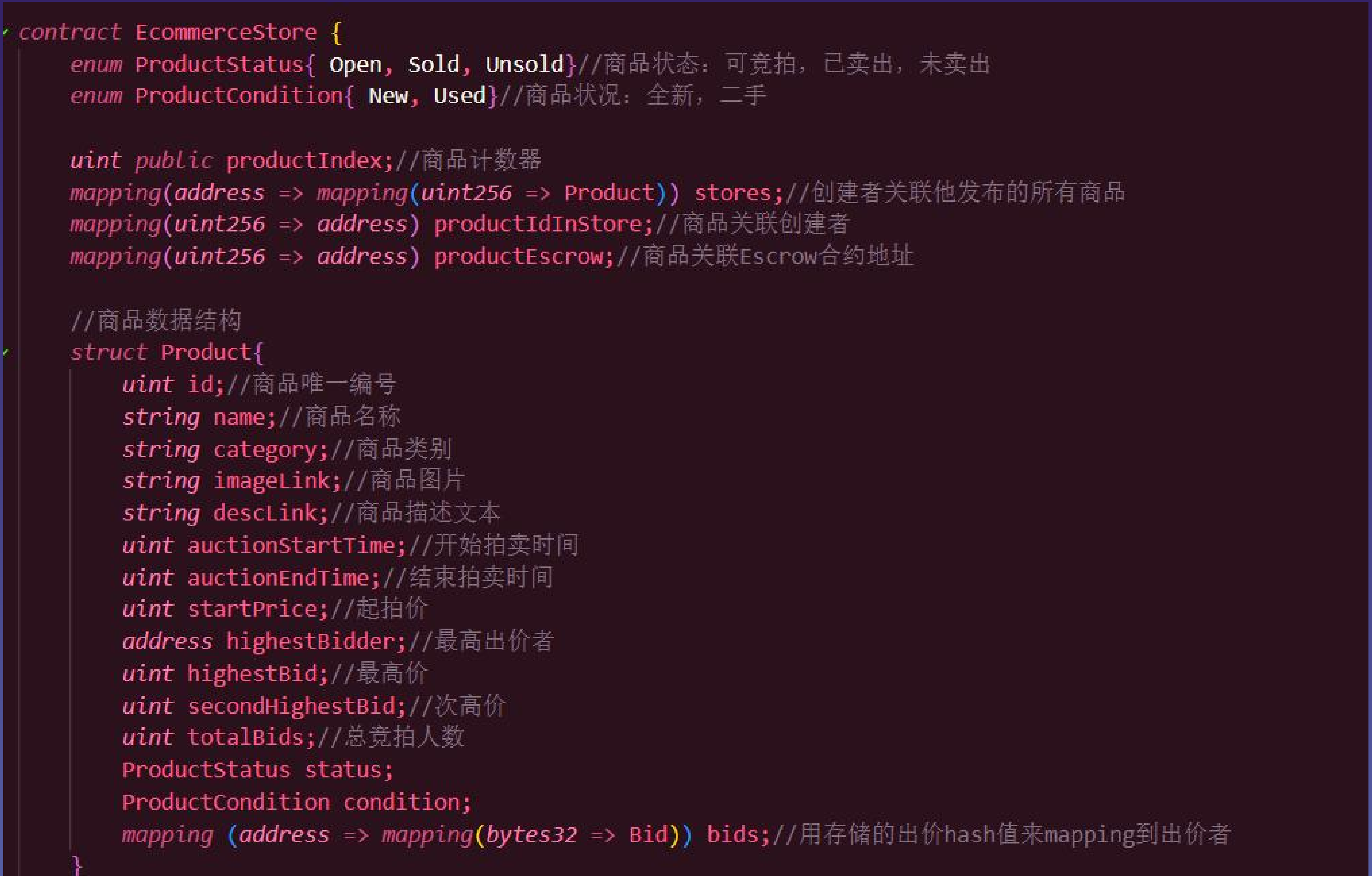
all

Auction Start Time

年 /月/日 --:--

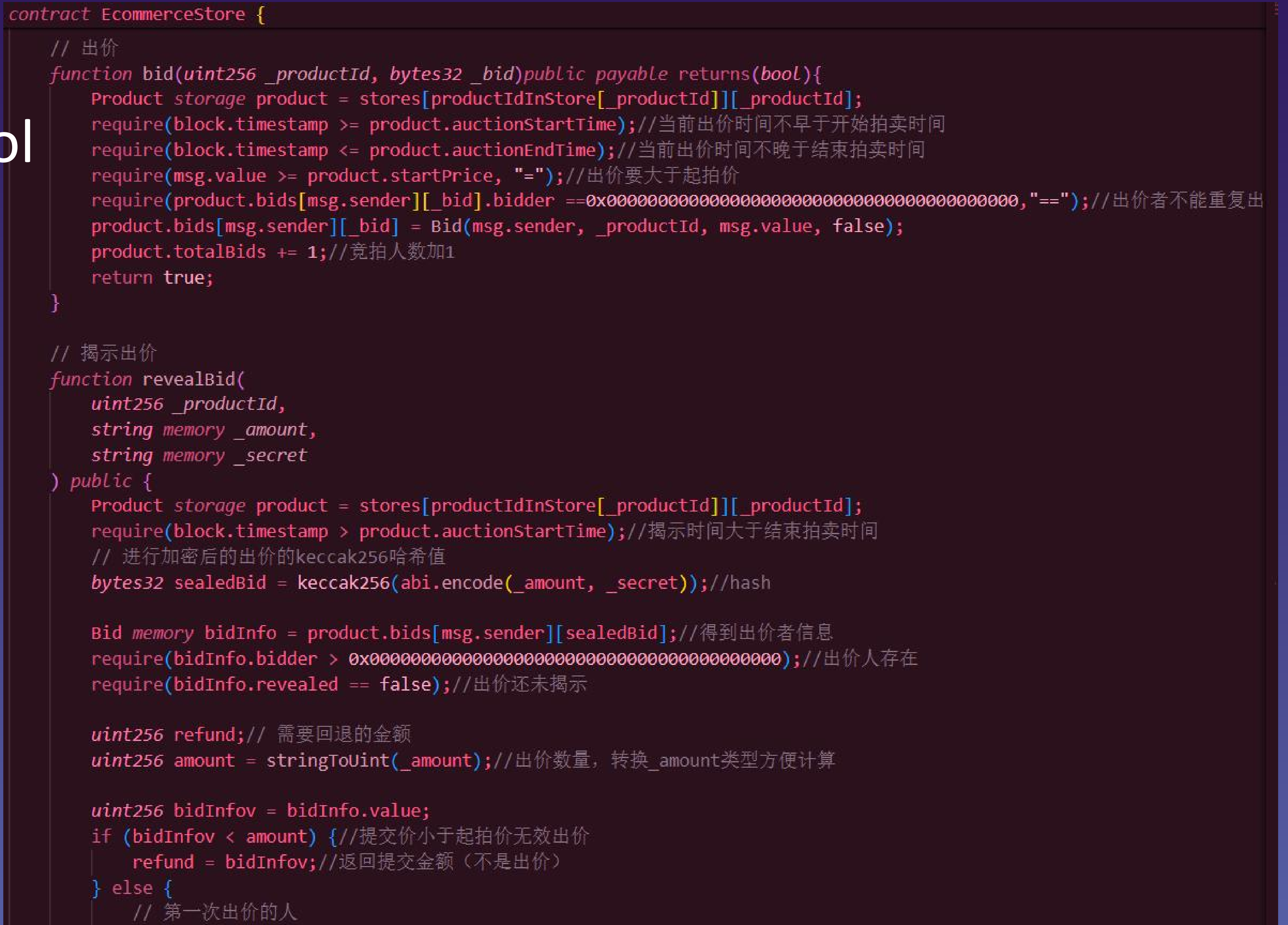
submit

智能合约Ecommerce.sol



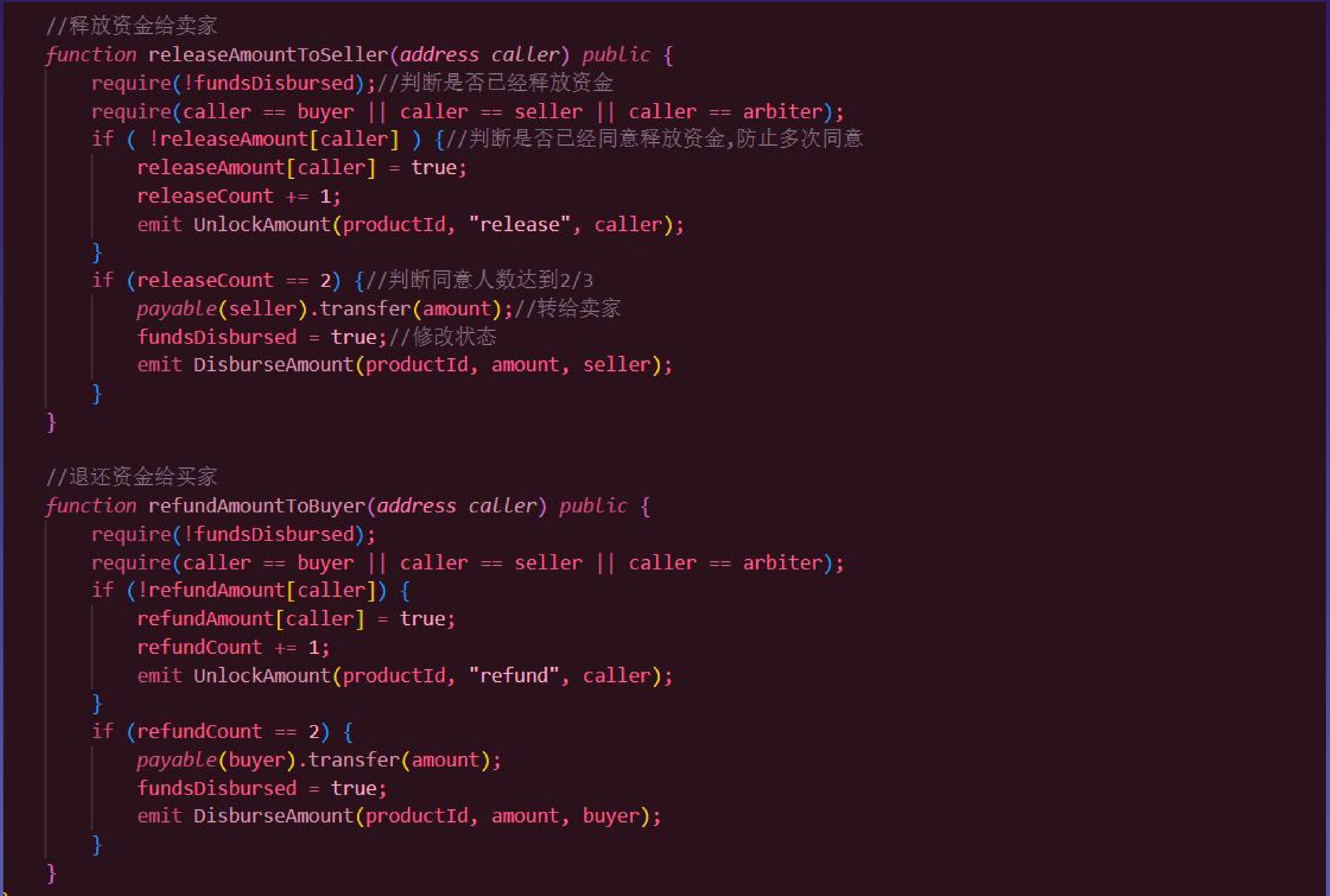
```
contract EcommerceStore {
    enum ProductStatus{ Open, Sold, Unsold}//商品状态：可竞拍，已卖出，未卖出
    enum ProductCondition{ New, Used}//商品状况：全新，二手

    uint public productIndex;//商品计数器
    mapping(address => mapping(uint256 => Product)) stores;//创建者关联他发布的所有商品
    mapping(uint256 => address) productIdInStore;//商品关联创建者
    mapping(uint256 => address) productEscrow;//商品关联Escrow合约地址

    //商品数据结构
    struct Product{
        uint id;//商品唯一编号
        string name;//商品名称
        string category;//商品类别
        string imageLink;//商品图片
        string descLink;//商品描述文本
        uint auctionStartTime;//开始拍卖时间
        uint auctionEndTime;//结束拍卖时间
        uint startPrice;//起拍价
        address highestBidder;//最高出价者
        uint highestBid;//最高价
        uint secondHighestBid;//次高价
        uint totalBids;//总竞拍人数
        ProductStatus status;
        ProductCondition condition;
        mapping (address => mapping(bytes32 => Bid)) bids;//用存储的出价hash值来mapping到出价者
    }
```

智能合约
Ecommerce.sol



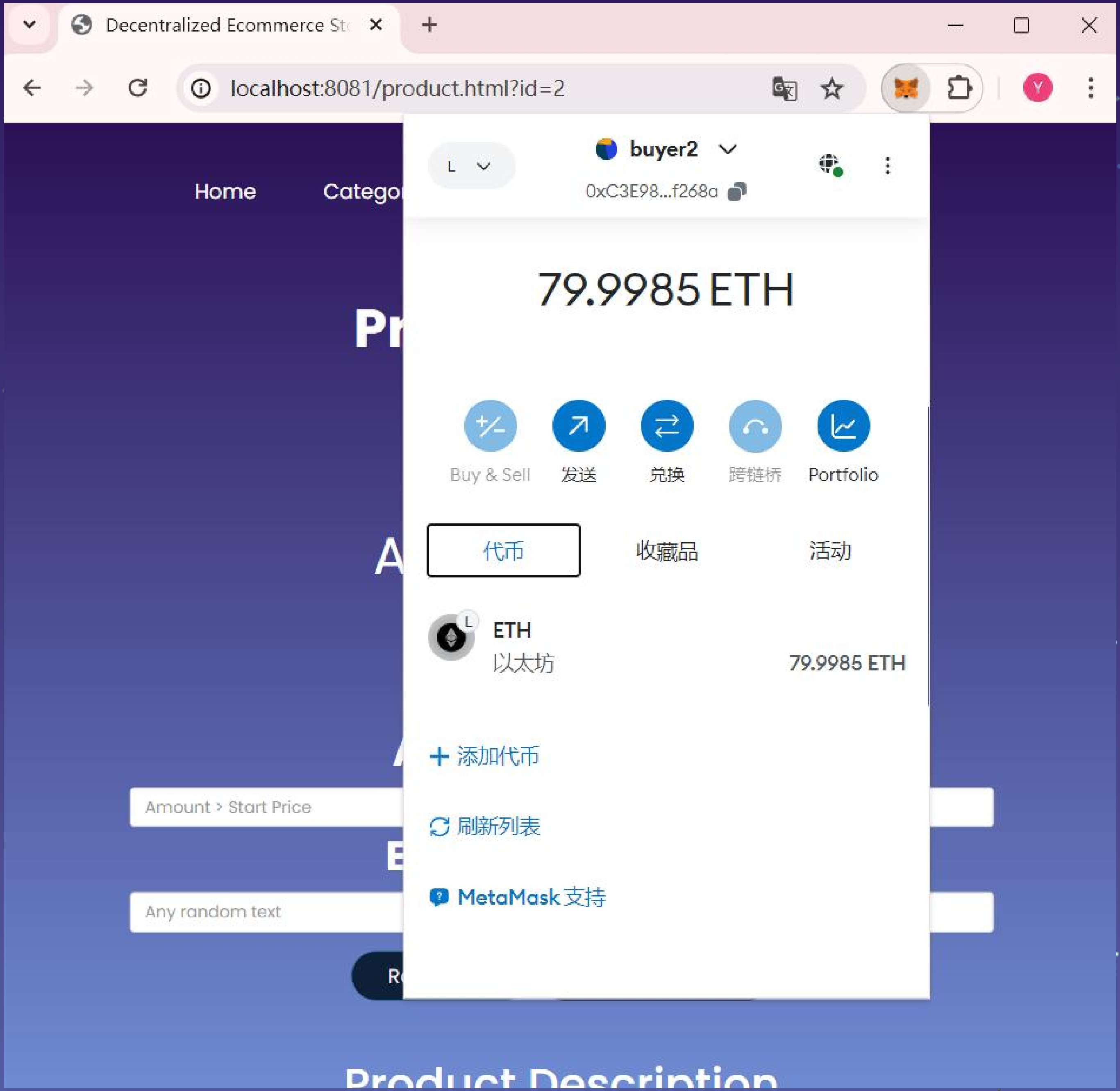
```
contract EcommerceStore {
    // 出价
    function bid(uint256 _productId, bytes32 _bid)public payable returns(bool){
        Product storage product = stores[productIdInStore[_productId]][_productId];
        require(block.timestamp >= product.auctionStartTime);//当前出价时间不早于开始拍卖时间
        require(block.timestamp <= product.auctionEndTime);//当前出价时间不晚于结束拍卖时间
        require(msg.value >= product.startPrice, "=");//出价要大于起拍价
        require(product.bids[msg.sender][_bid].bidder ==0x0000000000000000000000000000000000000000,"==");//出价者不能重复出
        product.bids[msg.sender][_bid] = Bid(msg.sender, _productId, msg.value, false);
        product.totalBids += 1;//竞拍人数加1
        return true;
    }

    // 揭示出价
    function revealBid(
        uint256 _productId,
        string memory _amount,
        string memory _secret
    ) public {
        Product storage product = stores[productIdInStore[_productId]][_productId];
        require(block.timestamp > product.auctionStartTime);//揭示时间大于结束拍卖时间
        // 进行加密后的出价的keccak256哈希值
        bytes32 sealedBid = keccak256(abi.encode(_amount, _secret));//hash

        Bid memory bidInfo = product.bids[msg.sender][sealedBid];//得到出价者信息
        require(bidInfo.bidder > 0x0000000000000000000000000000000000000000);//出价人存在
        require(bidInfo.revealed == false);//出价还未揭示

        uint256 refund;// 需要回退的金额
        uint256 amount = stringToUint(_amount);//出价数量，转换_amount类型方便计算

        uint256 bidInfov = bidInfo.value;
        if (bidInfov < amount) {//提交价小于起拍价无效出价
            refund = bidInfov;//返回提交金额（不是出价）
        } else {
            // 第一次出价的人
```

智能合约
Escrow.sol

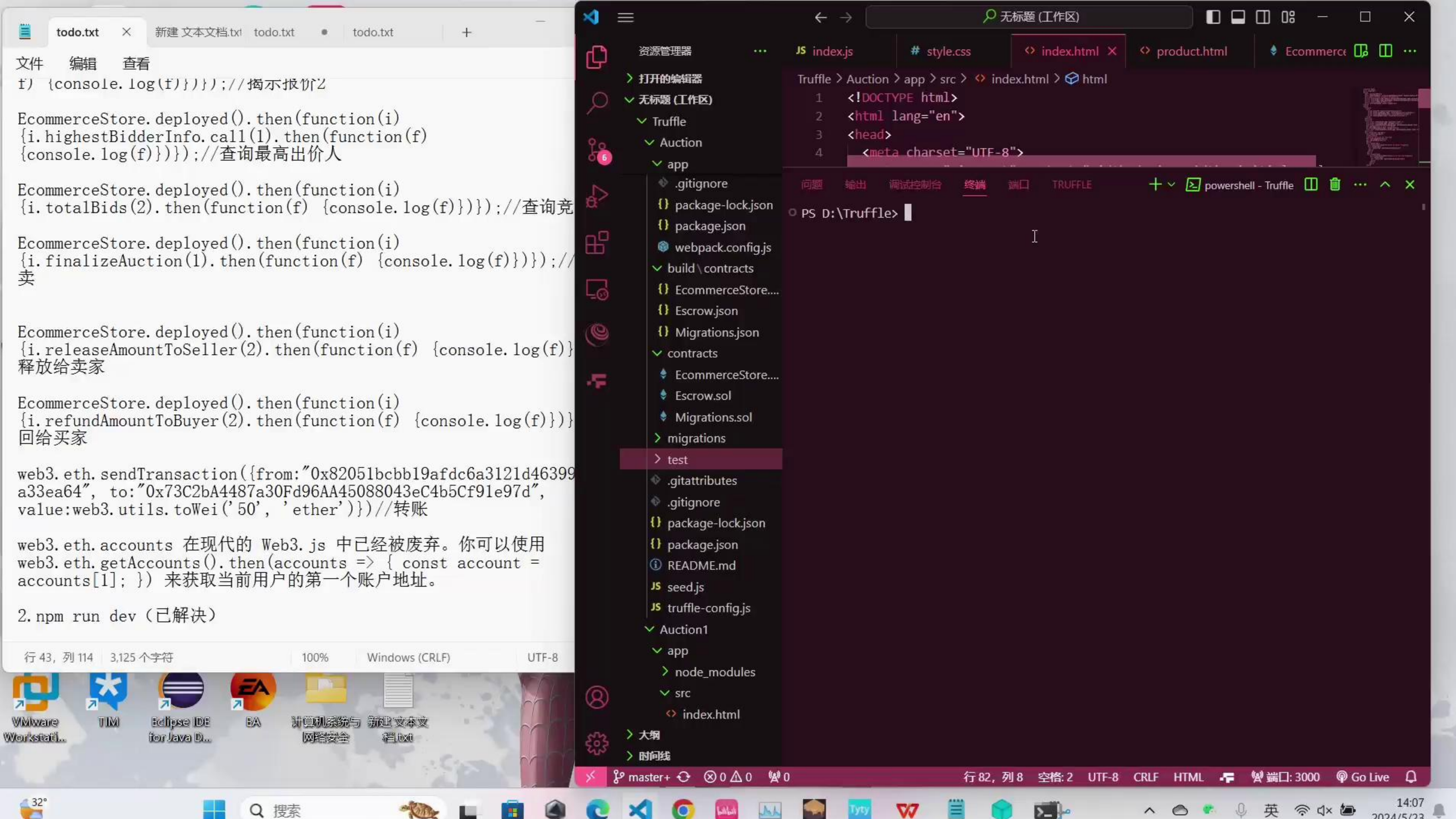


```
//释放资金给卖家
function releaseAmountToSeller(address caller) public {
    require(!fundsDisbursed);//判断是否已经释放资金
    require(caller == buyer || caller == seller || caller == arbiter);
    if ( !releaseAmount[caller] ) {//判断是否已经同意释放资金,防止多次同意
        releaseAmount[caller] = true;
        releaseCount += 1;
        emit UnlockAmount(productId, "release", caller);
    }
    if (releaseCount == 2) {//判断同意人数达到2/3
        payable(seller).transfer(amount);//转给卖家
        fundsDisbursed = true;//修改状态
        emit DisburseAmount(productId, amount, seller);
    }
}

//退还资金给买家
function refundAmountToBuyer(address caller) public {
    require(!fundsDisbursed);
    require(caller == buyer || caller == seller || caller == arbiter);
    if (!refundAmount[caller]) {
        refundAmount[caller] = true;
        refundCount += 1;
        emit UnlockAmount(productId, "refund", caller);
    }
    if (refundCount == 2) {
        payable(buyer).transfer(amount);
        fundsDisbursed = true;
        emit DisburseAmount(productId, amount, buyer);
    }
}
```

用MetaMask支付



目录
Direction



```
f) {console.log(f)}})});//揭示报价2

EcommerceStore.deployed().then(function(i)
{i.highestBidderInfo.call(1).then(function(f)
{console.log(f)}})});//查询最高出价人

EcommerceStore.deployed().then(function(i)
{i.totalBids(2).then(function(f) {console.log(f)}})});//查询竞

EcommerceStore.deployed().then(function(i)
{i.finalizeAuction(1).then(function(f) {console.log(f)}})});//
卖


EcommerceStore.deployed().then(function(i)
{i.releaseAmountToSeller(2).then(function(f) {console.log(f)}
释放给卖家

EcommerceStore.deployed().then(function(i)
{i.refundAmountToBuyer(2).then(function(f) {console.log(f)})}
回给买家

web3.eth.sendTransaction({from:"0x82051bcbb19afdc6a3121d46399
a33ea64", to:"0x73C2bA4487a30Fd96AA45088043eC4b5Cf91e97d",
value:web3.utils.toWei('50', 'ether')})//转账

web3.eth.accounts 在现代的 Web3.js 中已经被废弃。你可以使用
web3.eth.getAccounts().then(accounts => { const account =
accounts[1]; }) 来获取当前用户的第一个账户地址。

2.npm run dev（已解决）
```

Truffle > Auction > app > src > index.html > html

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
```

```
PS D:\Truffle>
```

Thank you

汇报人：刘伊蕾