Kevin Tran

Professor Maiti

CS5833

28 April 2024

Blockchains Project Report

Project Design:

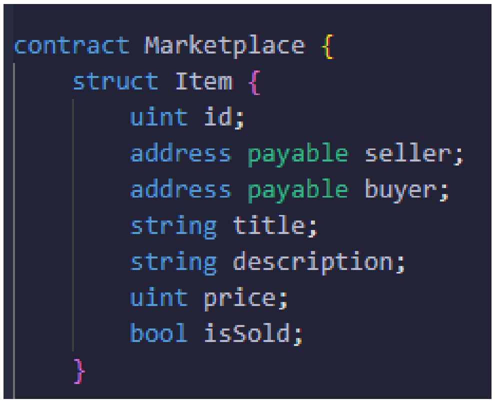
Starting with the contract, I wanted the seller to have the ability to list items with a title, description, price, and know the buyer's information. In my contract, the seller must at least specify the price to be at least 1 wei or more. If an item was listed from the seller, I wanted the buyer to be able to know the title, description, price of the item, the availability of the item, and the seller's information. In my contract, I made sure that the transaction will not go through if the buyer tries to buy an item that they listed, purchase an item that was sold, or if the buyer does not pay the amount of the item's price. In my item object/struct, I decided to include an id to keep track of the item, an address of the seller, an address of the buyer, item title, item description, item price, and a boolean flag to indicate the availability of the item. Upon visiting the website, the website will check if there is a MetaMask browser extension and if a MetaMask wallet has been connected to the website. If a MetaMask wallet has not been connected, the website will prompt to connect an account. Additionally, the website will display all the items, including both sold and available items, on the marketplace. The items displayed will include information such as item title, description, value, the buyer's address, the seller's address, and status of whether the item is available for purchase. If an item is available for purchase, the listed item will display a purchase button that a user can click to purchase the item. If an item has already been purchased, the website will just display it, but does not have a purchase button. Instead of using Google Cloud Platform to host the website, I asked Dr. Maiti using GitHub pages to host the website and he said to go for it.

GitHub repo link: **https://github.com/19kevintran/CS5833Project/**

Marketplace link: **https://19kevintran.github.io/CS5833Project/**

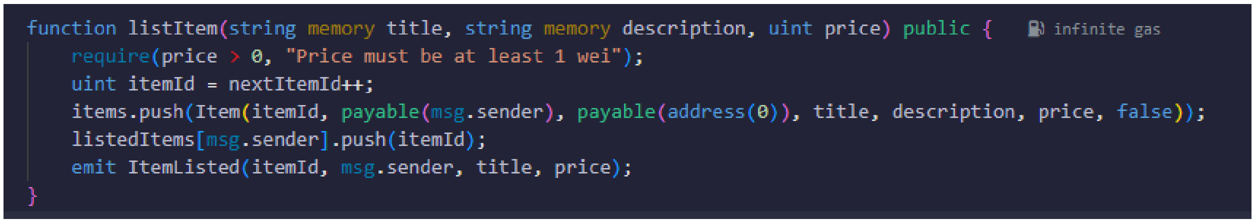
Smart contract logic

Struct of item: Item includes an id to keep track of the item, an address of the seller, an address of the buyer, item title, item description, item price, and a boolean flag to indicate the availability of the item.



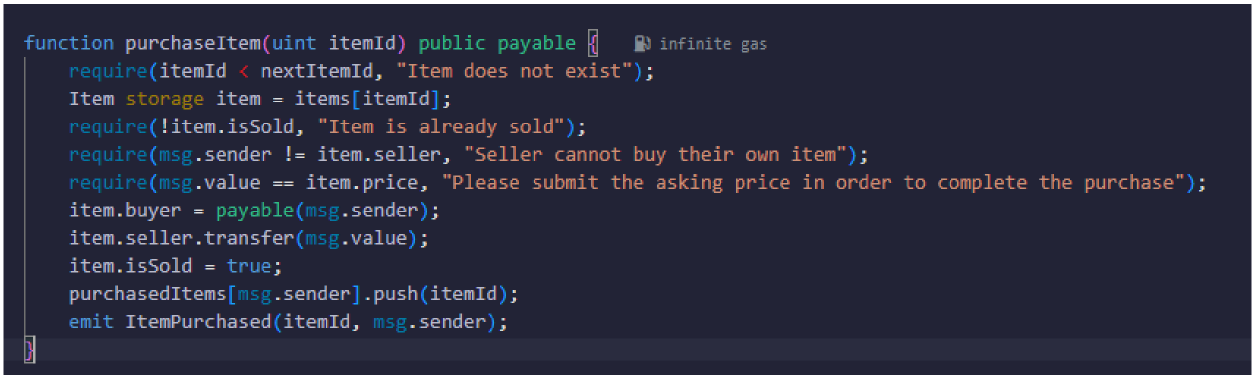
```
contract Marketplace {
    struct Item {
        uint id;
        address payable seller;
        address payable buyer;
        string title;
        string description;
        uint price;
        bool isSold;
    }
```

listItem function: Seller must list title, description, and price of the item. Item id automatically mapped to the id.



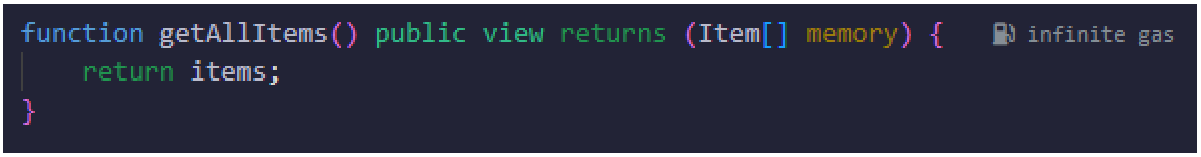
```
function listItem(string memory title, string memory description, uint price) public {    infinite gas
    require(price > 0, "Price must be at least 1 wei");
    uint itemId = nextItemId++;
    items.push(Item(itemId, payable(msg.sender), payable(address(0)), title, description, price, false));
    listedItems[msg.sender].push(itemId);
    emit ItemListed(itemId, msg.sender, title, price);
}
```

purchaseItem function: Buyer buys an item via item id.



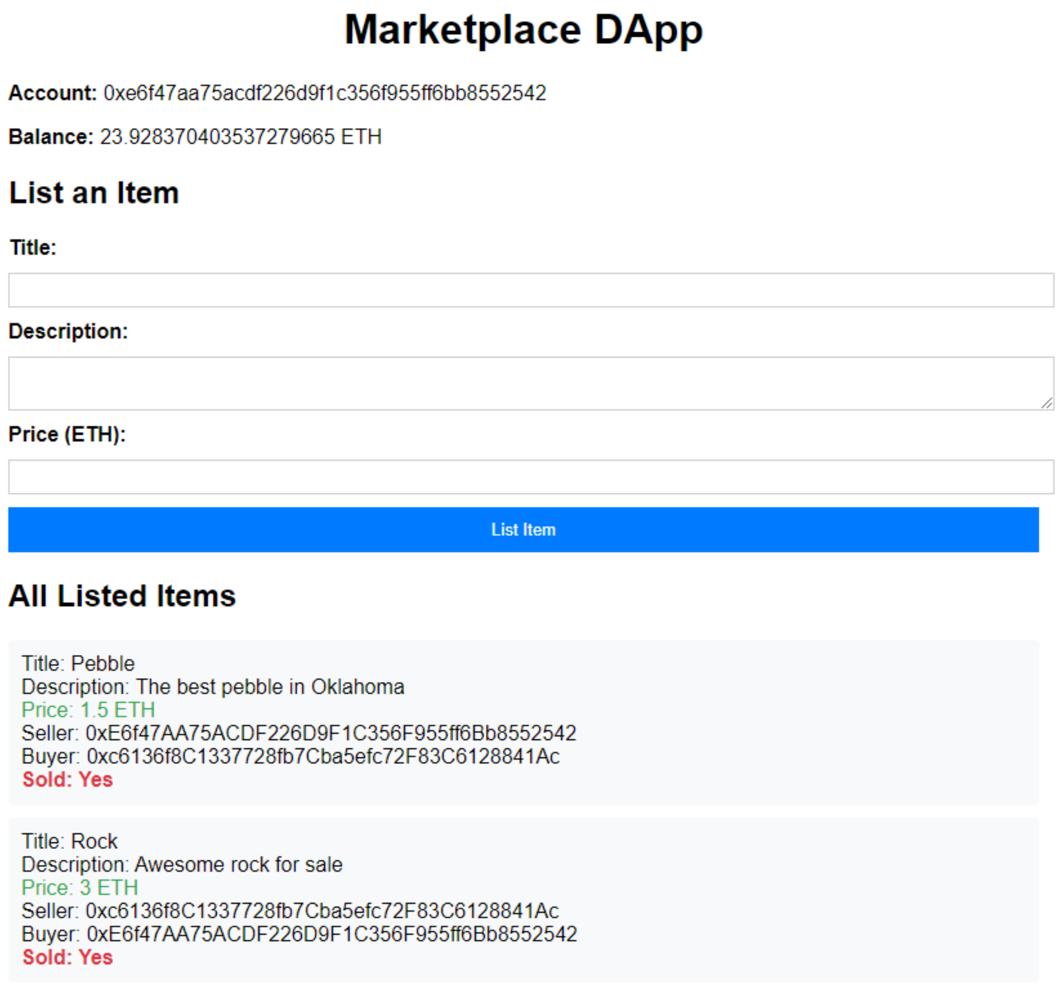
```
function purchaseItem(uint itemId) public payable {    infinite gas
    require(itemId < nextItemId, "Item does not exist");
    Item storage item = items[itemId];
    require(!item.isSold, "Item is already sold");
    require(msg.sender != item.seller, "Seller cannot buy their own item");
    require(msg.value == item.price, "Please submit the asking price in order to complete the purchase");
    item.buyer = payable(msg.sender);
    item.seller.transfer(msg.value);
    item.isSold = true;
    purchasedItems[msg.sender].push(itemId);
    emit ItemPurchased(itemId, msg.sender);
}
```

getaAllItems function: Returns all items and includes both sold and available items information (an address of the seller, an address of the buyer, item title, item description, item price, and a boolean flag to indicate the availability of the item)



```
function getAllItems() public view returns (Item[] memory) {    infinite gas
    return items;
}
```

Website User Interface:



Marketplace DApp

Account: 0xe6f47aa75acdf226d9f1c356f955ff6bb8552542

Balance: 23.928370403537279665 ETH

List an Item

Title:

Description:

Price (ETH):

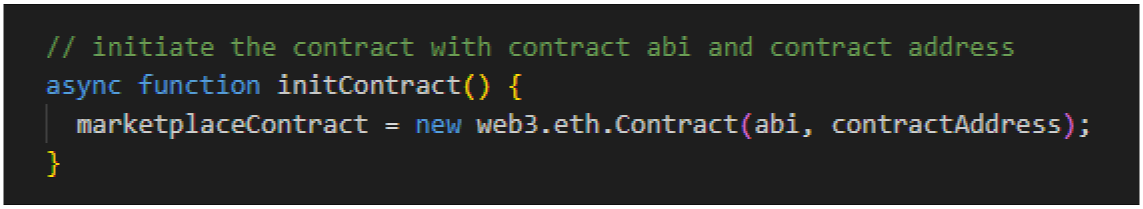
All Listed Items

Title: Pebble
Description: The best pebble in Oklahoma
Price: 1.5 ETH
Seller: 0xE6f47AA75ACDF226D9F1C356F955ff6Bb8552542
Buyer: 0xc6136f8C1337728fb7Cba5efc72F83C6128841Ac
Sold: Yes

Title: Rock
Description: Awesome rock for sale
Price: 3 ETH
Seller: 0xc6136f8C1337728fb7Cba5efc72F83C6128841Ac
Buyer: 0xE6f47AA75ACDF226D9F1C356F955ff6Bb8552542
Sold: Yes

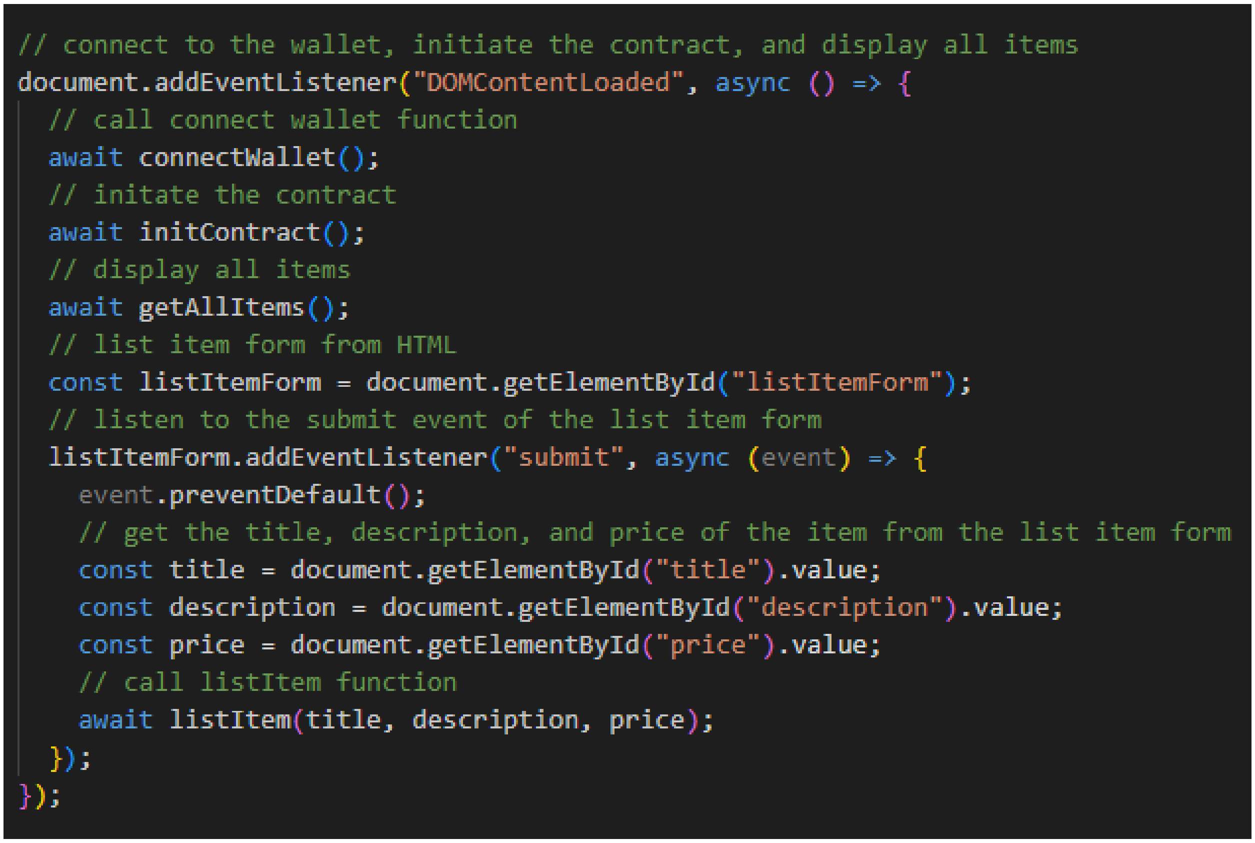
App.js functionality:

When the user goes to the website, app.js will first initiate the contract with the contract address and ABI.

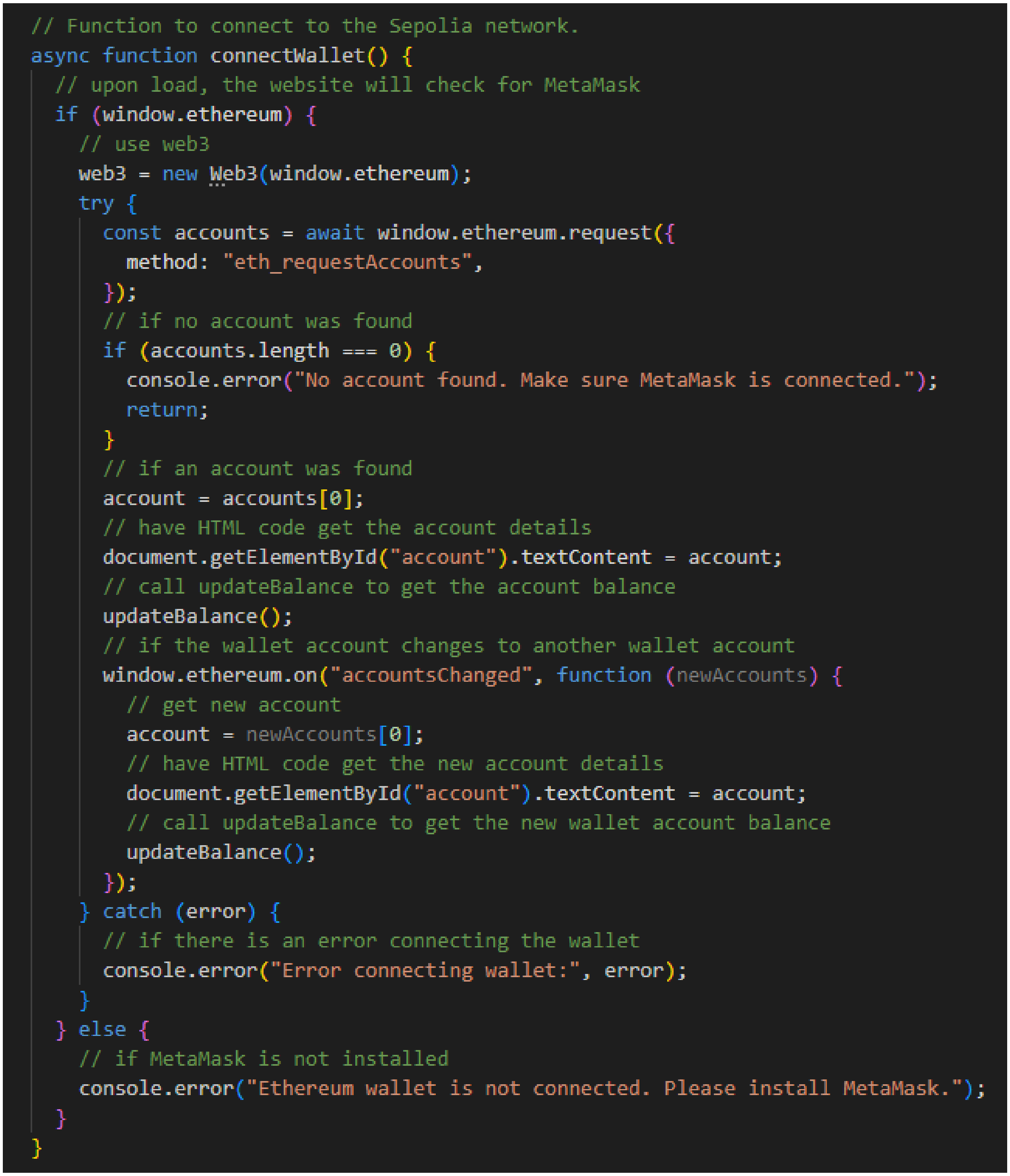


```
// initiate the contract with contract abi and contract address
async function initContract() {
    marketplaceContract = new web3.eth.Contract(abi, contractAddress);
}
```

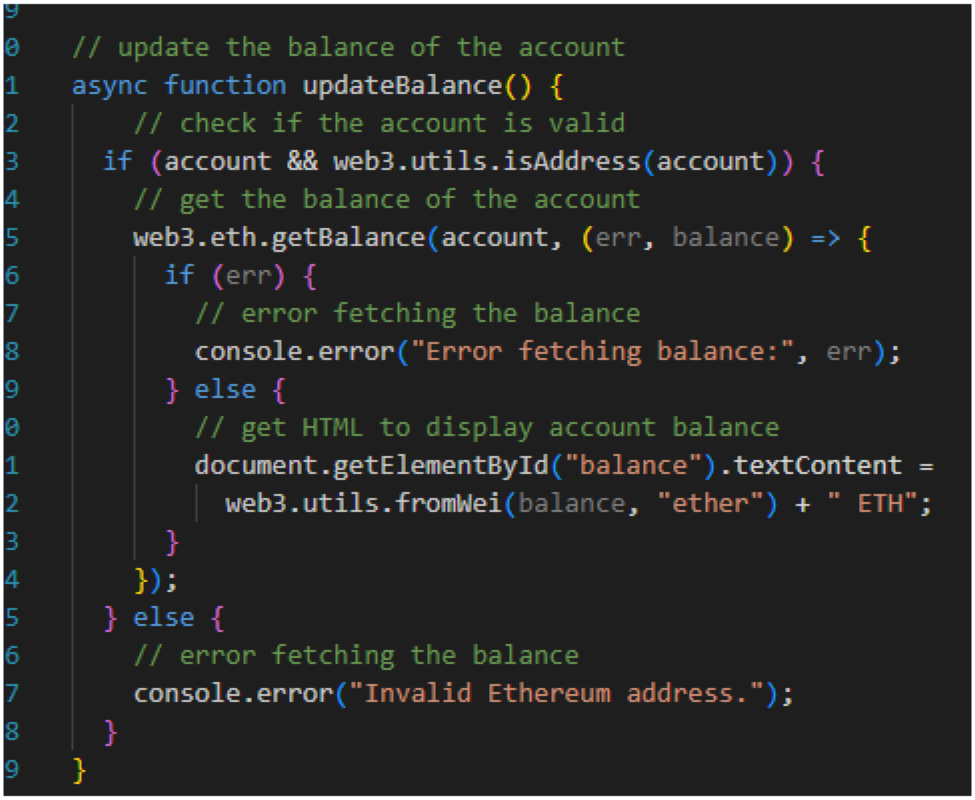
Upon load, the connectWallet function will look for MetaMask. If the user does not have MetaMask, it will throw an error stating that there is no connection and to install MetaMask. If the user has MetaMask, the connectWallet will wait until a user connects a MetaMask wallet. If the operation is not successful, it will throw an error connecting to the wallet. If successful, the wallet address and balance will appear. If a user chooses to sign in with a new wallet address, the account wallet address will update, call updateBalance function, and display the new wallet balance. In updateBalance, it will call the contract’s getBalance function which will return the balance of the address.



```
// connect to the wallet, initiate the contract, and display all items
document.addEventListener("DOMContentLoaded", async () => {
  // call connect wallet function
  await connectWallet();
  // initate the contract
  await initContract();
  // display all items
  await getAllItems();
  // list item form from HTML
  const listItemForm = document.getElementById("listItemForm");
  // listen to the submit event of the list item form
  listItemForm.addEventListener("submit", async (event) => {
    event.preventDefault();
    // get the title, description, and price of the item from the list item form
    const title = document.getElementById("title").value;
    const description = document.getElementById("description").value;
    const price = document.getElementById("price").value;
    // call listItem function
    await listItem(title, description, price);
  });
});
```

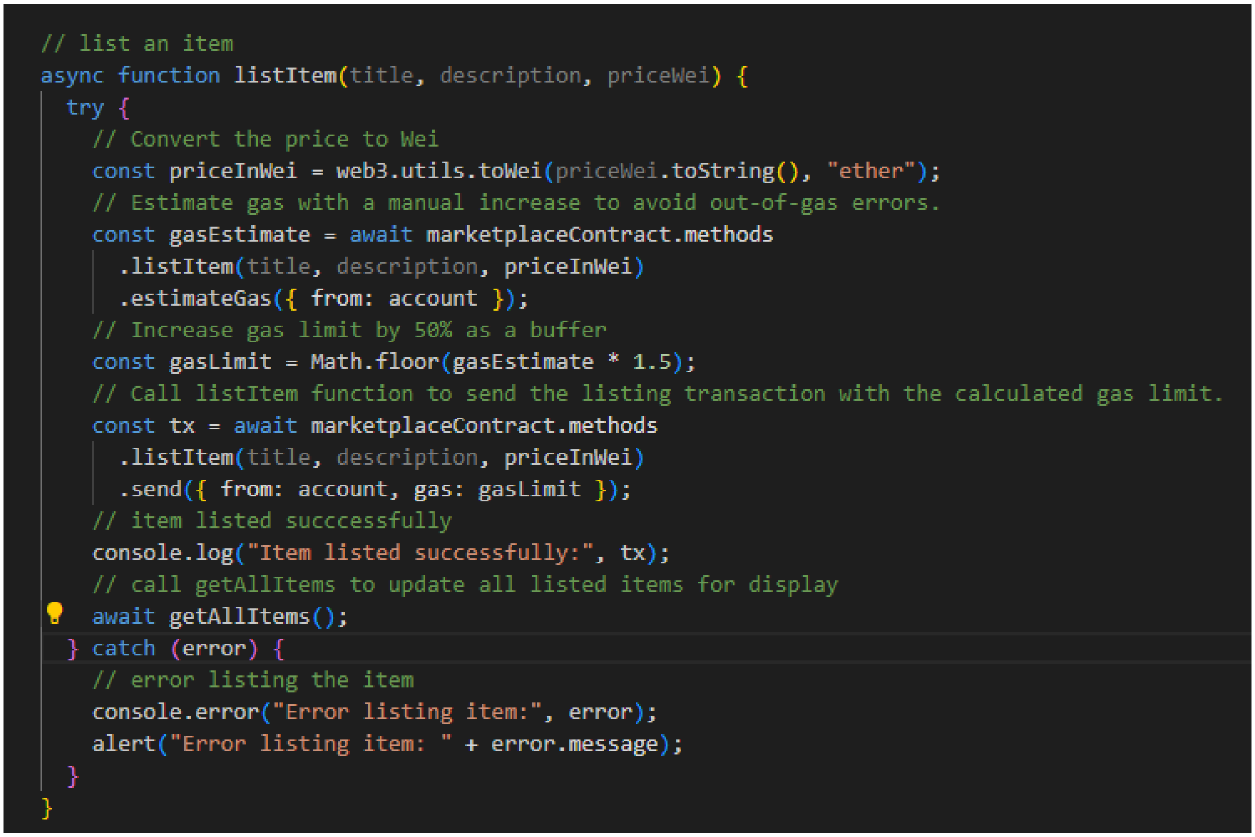


```
// Function to connect to the Sepolia network.
async function connectWallet() {
  // upon load, the website will check for MetaMask
  if (window.ethereum) {
    // use web3
    web3 = new Web3(window.ethereum);
    try {
      const accounts = await window.ethereum.request({
        method: "eth_requestAccounts",
      });
      // if no account was found
      if (accounts.length === 0) {
        console.error("No account found. Make sure MetaMask is connected.");
        return;
      }
      // if an account was found
      account = accounts[0];
      // have HTML code get the account details
      document.getElementById("account").textContent = account;
      // call updateBalance to get the account balance
      updateBalance();
      // if the wallet account changes to another wallet account
      window.ethereum.on("accountsChanged", function (newAccounts) {
        // get new account
        account = newAccounts[0];
        // have HTML code get the new account details
        document.getElementById("account").textContent = account;
        // call updateBalance to get the new wallet account balance
        updateBalance();
      });
    } catch (error) {
      // if there is an error connecting the wallet
      console.error("Error connecting wallet:", error);
    }
  } else {
    // if MetaMask is not installed
    console.error("Ethereum wallet is not connected. Please install MetaMask.");
  }
}
```



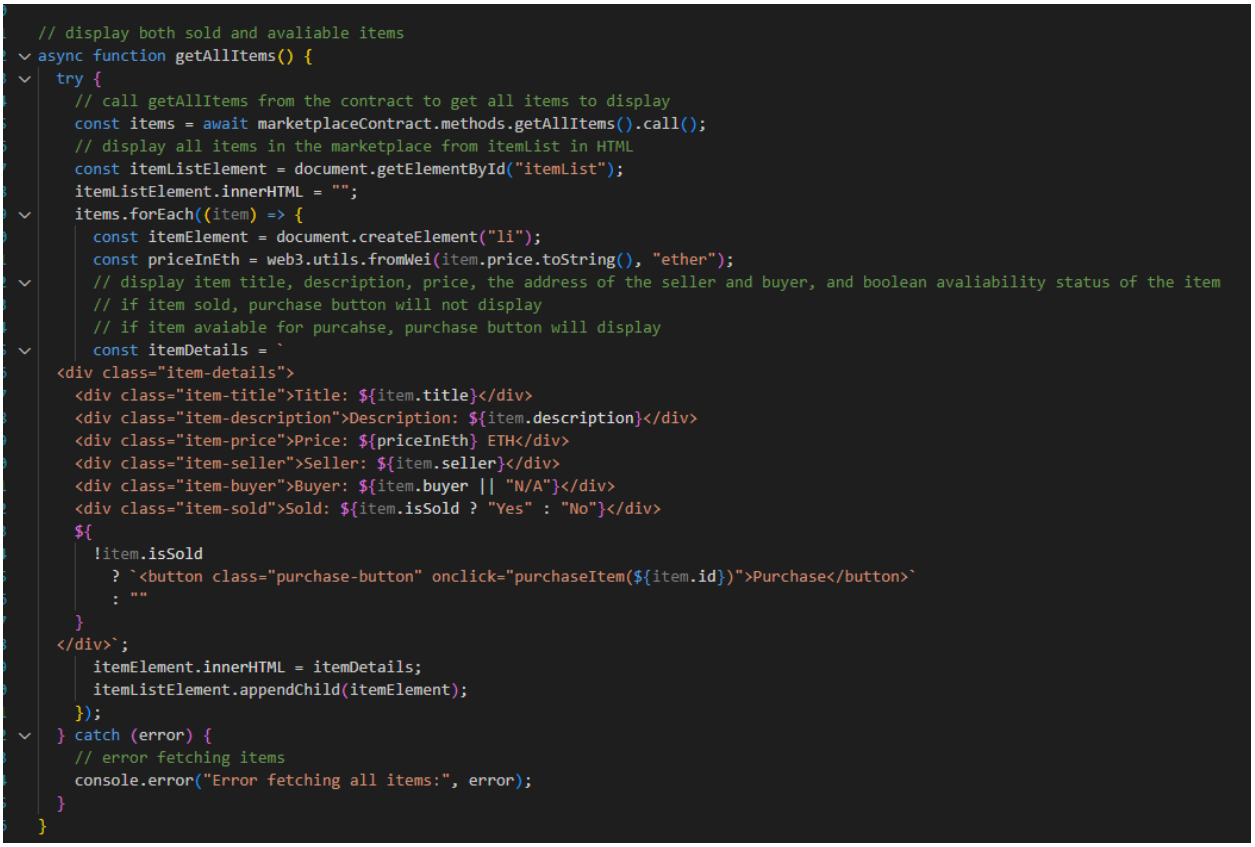
```javascript
// update the balance of the account
async function updateBalance() {
    // check if the account is valid
  if (account && web3.utils.isAddress(account)) {
    // get the balance of the account
    web3.eth.getBalance(account, (err, balance) => {
      if (err) {
        // error fetching the balance
        console.error("Error fetching balance:", err);
      } else {
        // get HTML to display account balance
        document.getElementById("balance").textContent =
          web3.utils.fromWei(balance, "ether") + " ETH";
      }
    });
  } else {
    // error fetching the balance
    console.error("Invalid Ethereum address.");
  }
}
```

When the user successfully signs in with their wallet address, they can list an item or purchase an available item. To list an item, the user will specify the item’s title, description, and value in the parameter fields. The listItem function will take those fields, calculate the gasLimit by multiplying the gasEstimate multiplied by 50%, and calling the contract’s listItem function with the title, description, and price of the item. If the item successfully lists, there will be a message stating that the listing transaction went through. If the item does not successfully list, it will throw an error. Towards the end of the function, it will call the getAllItems which will add the new listing to all listing items and display it via HTML.



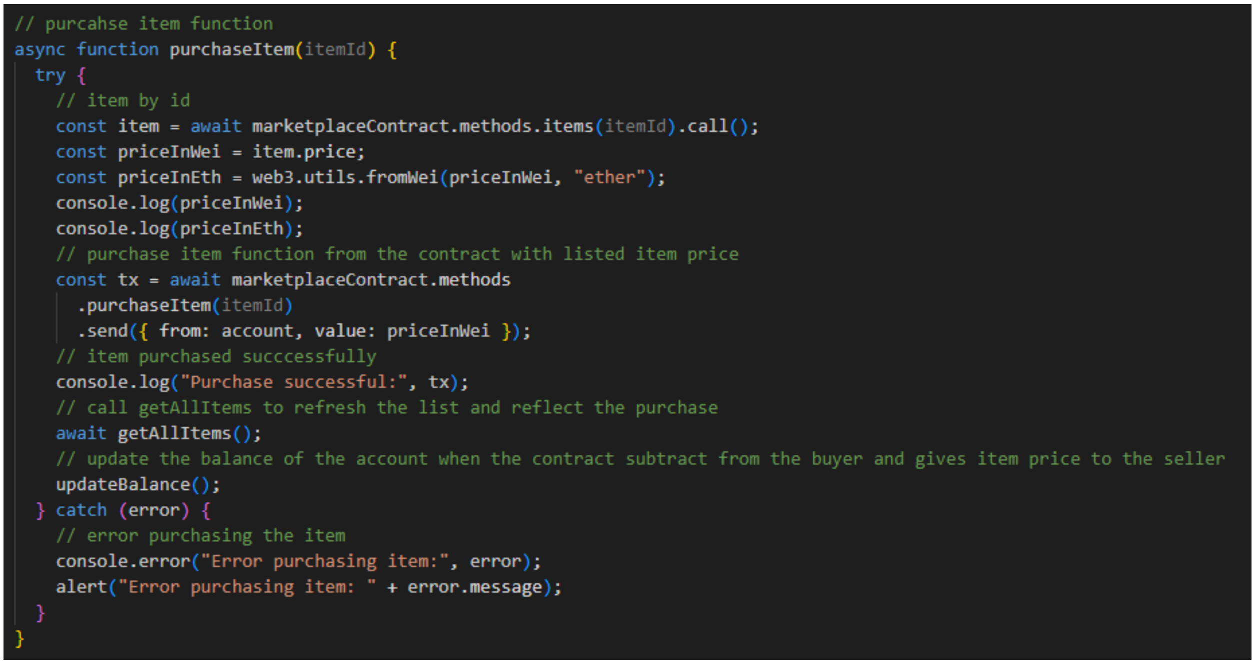
```
// list an item
async function listItem(title, description, priceWei) {
  try {
    // Convert the price to Wei
    const priceInWei = web3.utils.toWei(priceWei.toString(), "ether");
    // Estimate gas with a manual increase to avoid out-of-gas errors.
    const gasEstimate = await marketplaceContract.methods
      .listItem(title, description, priceInWei)
      .estimateGas({ from: account });
    // Increase gas limit by 50% as a buffer
    const gasLimit = Math.floor(gasEstimate * 1.5);
    // Call listItem function to send the listing transaction with the calculated gas limit.
    const tx = await marketplaceContract.methods
      .listItem(title, description, priceInWei)
      .send({ from: account, gas: gasLimit });
    // item listed succcessfully
    console.log("Item listed successfully:", tx);
    // call getAllItems to update all listed items for display
    await getAllItems();
  } catch (error) {
    // error listing the item
    console.error("Error listing item:", error);
    alert("Error listing item: " + error.message);
  }
}
```

In getAllItems function, it will call the getAllItems function and retrieve all the items listed for sale. This includes both sold and available items. If the item is available for purchase, the item will have a purchase button. If the item has already been sold, it will not display the purchase item and display that the boolean value for isSold is Yes.



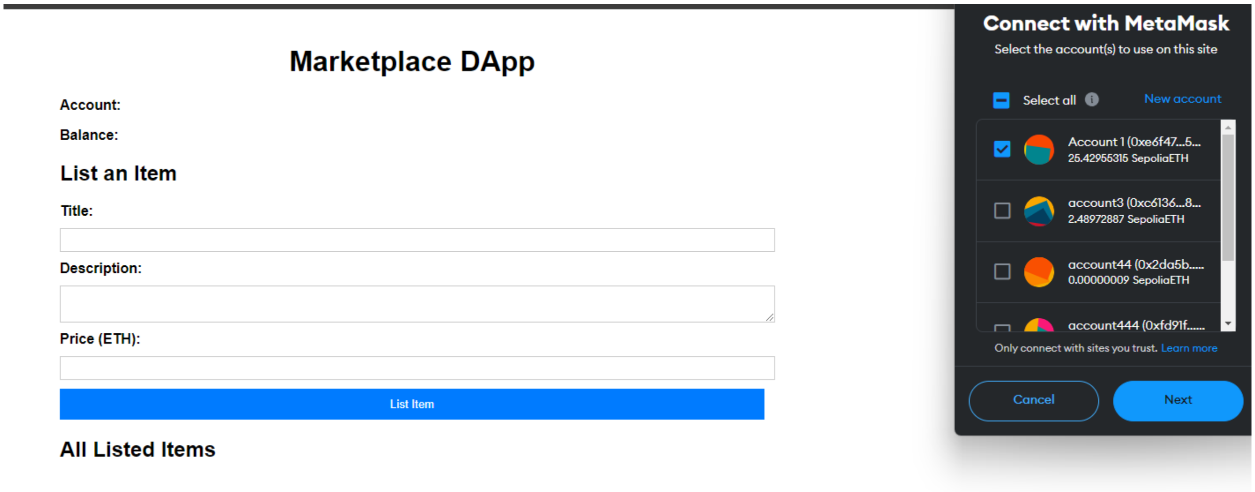
```
// display both sold and avaliable items
async function getAllItems() {
  try {
    // call getAllItems from the contract to get all items to display
    const items = await marketplaceContract.methods.getAllItems().call();
    // display all items in the marketplace from itemList in HTML
    const itemListElement = document.getElementById("itemList");
    itemListElement.innerHTML = "";
    items.forEach((item) => {
      const itemElement = document.createElement("li");
      const priceInEth = web3.utils.fromWei(item.price.toString(), "ether");
      // display item title, description, price, the address of the seller and buyer, and boolean avaliability status of the item
      // if item sold, purchase button will not display
      // if item avaiable for purcahse, purchase button will display
      const itemDetails = `
  <div class="item-details">
    <div class="item-title">Title: ${item.title}</div>
    <div class="item-description">Description: ${item.description}</div>
    <div class="item-price">Price: ${priceInEth} ETH</div>
    <div class="item-seller">Seller: ${item.seller}</div>
    <div class="item-buyer">Buyer: ${item.buyer || "N/A"}</div>
    <div class="item-sold">Sold: ${item.isSold ? "Yes" : "No"}</div>
    ${
      !item.isSold
        ? `<button class="purchase-button" onclick="purchaseItem(${item.id})">Purchase</button>`
        : ""
    }
  </div>`;
      itemElement.innerHTML = itemDetails;
      itemListElement.appendChild(itemElement);
    });
  } catch (error) {
    // error fetching items
    console.error("Error fetching all items:", error);
  }
}
```

Lastly, if the user chooses to purchase an available item, the purchaseItem function call the items function with the item id to retrieve the item’s price. After retrieving the item price, it will call the purchaseItem function in the contract with the itemId, buyer’s account, and payment value. If the purchase item transaction went through, there will be a success message. Towards the end of the purchaseItem function, it will call getAllItems to update the item’s listing to sold and pdate the buyer’s balance.



```
// purcahse item function
async function purchaseItem(itemId) {
  try {
    // item by id
    const item = await marketplaceContract.methods.items(itemId).call();
    const priceInWei = item.price;
    const priceInEth = web3.utils.fromWei(priceInWei, "ether");
    console.log(priceInWei);
    console.log(priceInEth);
    // purchase item function from the contract with listed item price
    const tx = await marketplaceContract.methods
      .purchaseItem(itemId)
      .send({ from: account, value: priceInWei });
    // item purchased succcessfully
    console.log("Purchase successful:", tx);
    // call getAllItems to refresh the list and reflect the purchase
    await getAllItems();
    // update the balance of the account when the contract subtract from the buyer and gives item price to the seller
    updateBalance();
  } catch (error) {
    // error purchasing the item
    console.error("Error purchasing item:", error);
    alert("Error purchasing item: " + error.message);
  }
}
```

<u>DEMO:</u> Entering into the website.



Marketplace DApp

Account:

Balance:

List an Item

Title:

Description:

Price (ETH):

All Listed Items

Connect with MetaMask

Select the account(s) to use on this site

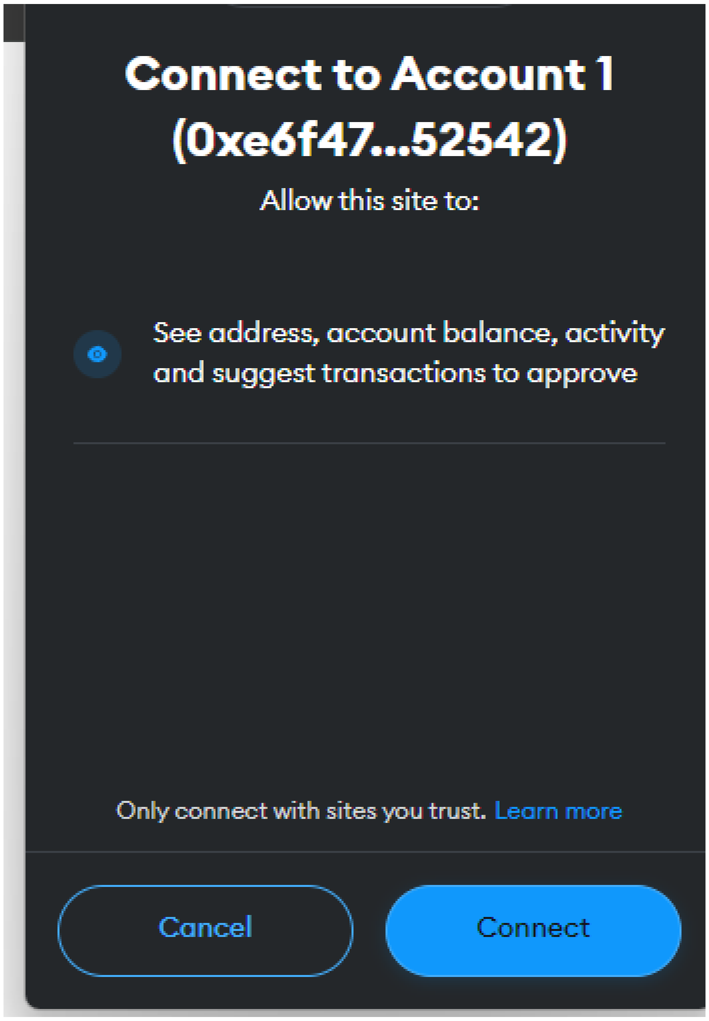
☑ Account 1 (0xe6f47...5... 25.42955315 SepoliaETH

□ account3 (0xc6136...8... 2.48972887 SepoliaETH

□ account44 (0x2da5b..... 0.00000009 SepoliaETH

□ account444 (0xfd91f......

Only connect with sites you trust. Learn more



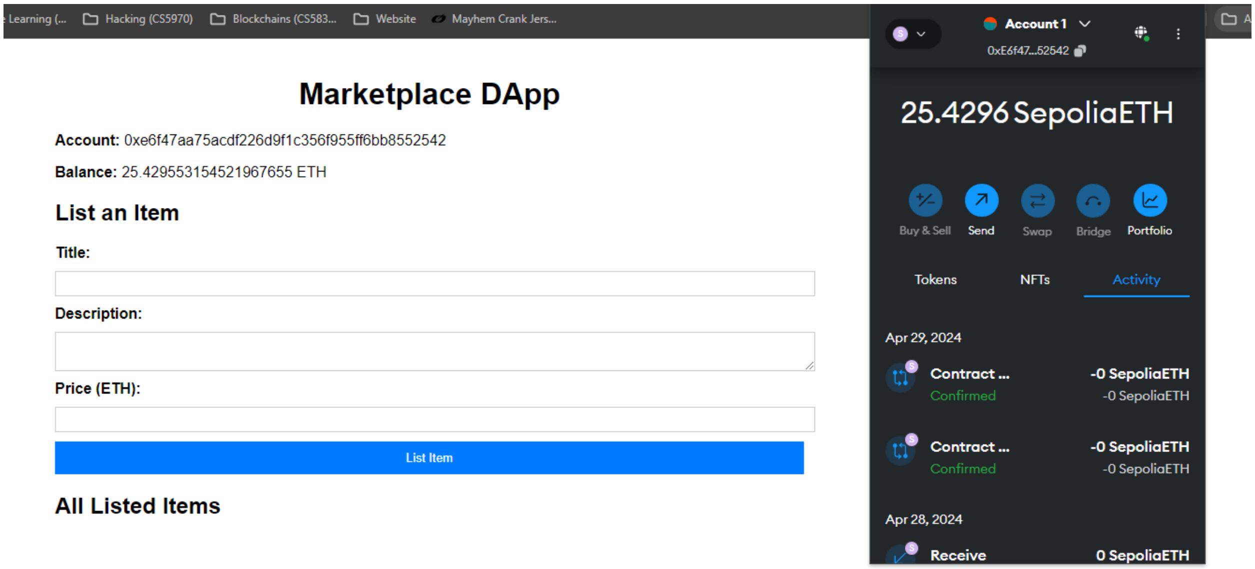
Connect to Account 1 (0xe6f47...52542)

Allow this site to:

See address, account balance, activity and suggest transactions to approve

Only connect with sites you trust. Learn more

Once a MetaMask wallet has been connected, it will display the user’s address and wallet balance.



Marketplace DApp

Account: 0xe6f47aa75acdf226d9f1c356f955ff6bb8552542

Balance: 25.429553154521967655 ETH

List an Item

Title:

Description:

Price (ETH):

All Listed Items

Account 1

0xE6f47...52542

25.4296 SepoliaETH

Apr 29, 2024

Contract ... -0 SepoliaETH
Confirmed -0 SepoliaETH

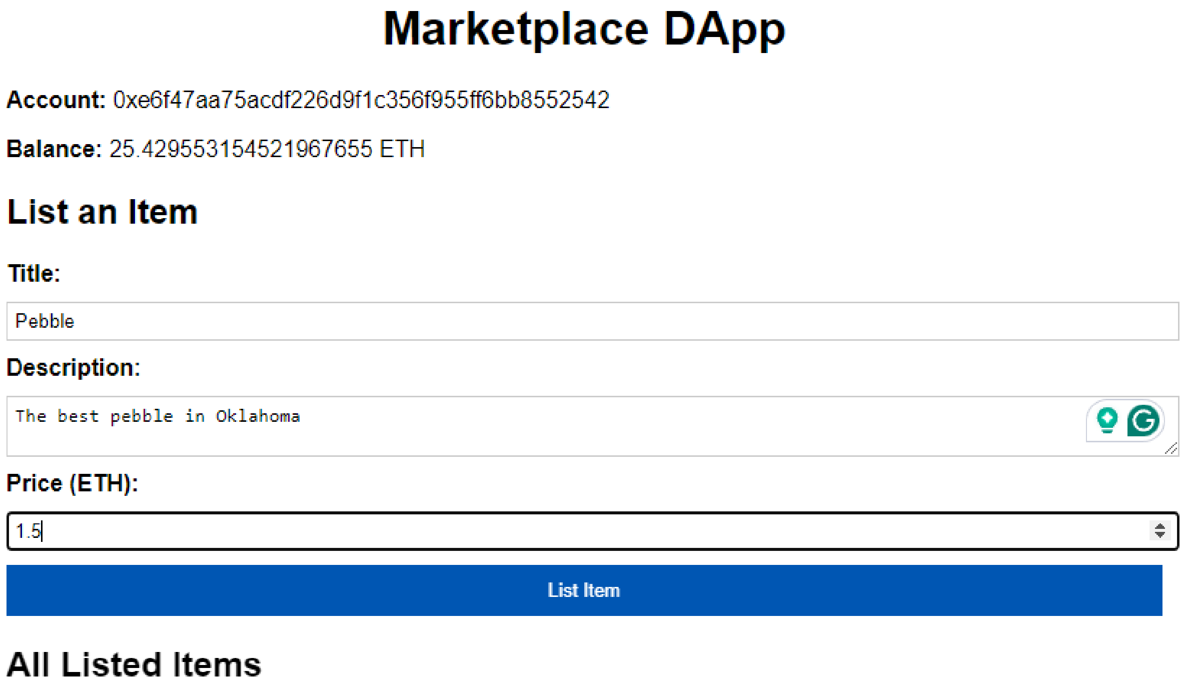
Contract ... -0 SepoliaETH
Confirmed -0 SepoliaETH

Apr 28, 2024

Receive 0 SepoliaETH

The user will have an option to list an item by providing the title, description, and item price.

Example:



Marketplace DApp

Account: 0xe6f47aa75acdf226d9f1c356f955ff6bb8552542

Balance: 25.429553154521967655 ETH

List an Item

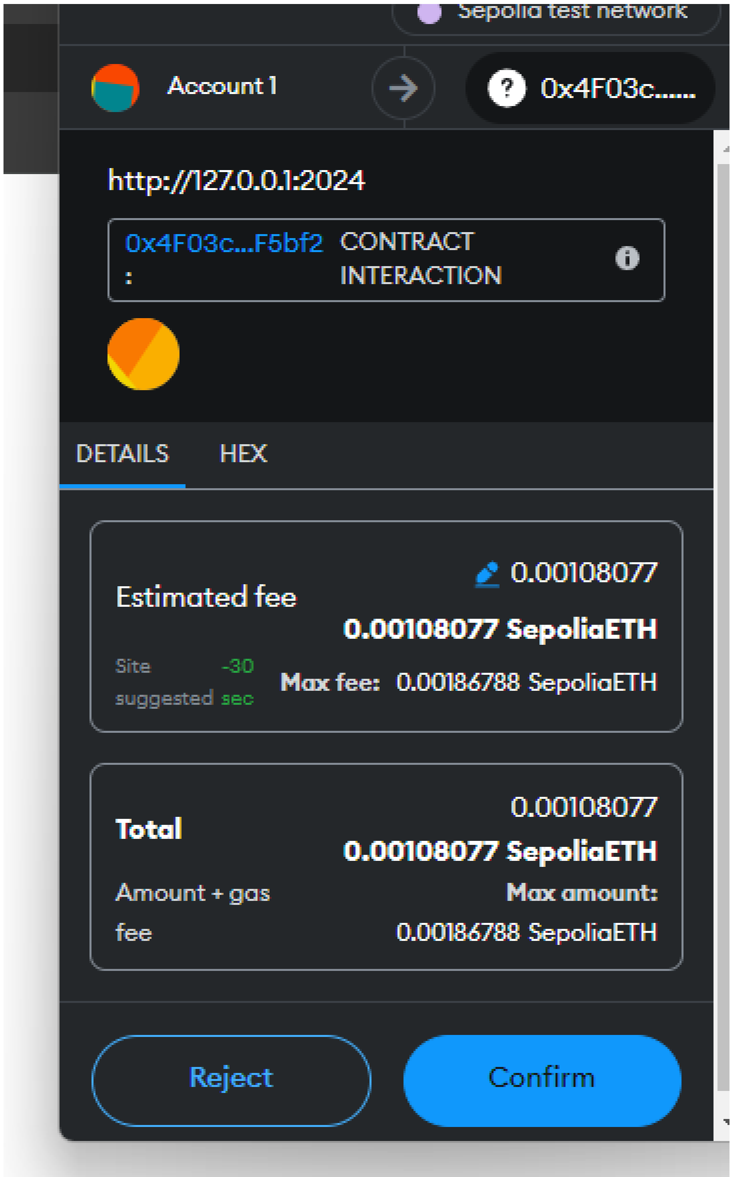
Title: Pebble

Description: The best pebble in Oklahoma

Price (ETH): 1.5

All Listed Items

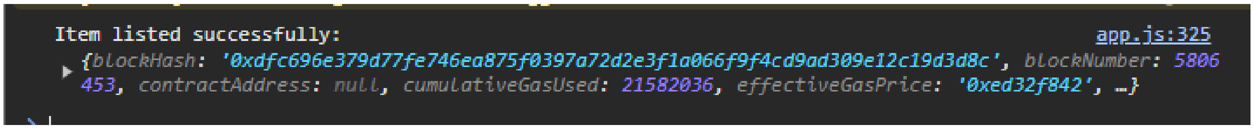
After clicking the List Item Button, it will ask for a transaction fee.



Estimated fee 0.00108077
0.00108077 SepoliaETH
Site suggested -30 sec Max fee: 0.00186788 SepoliaETH

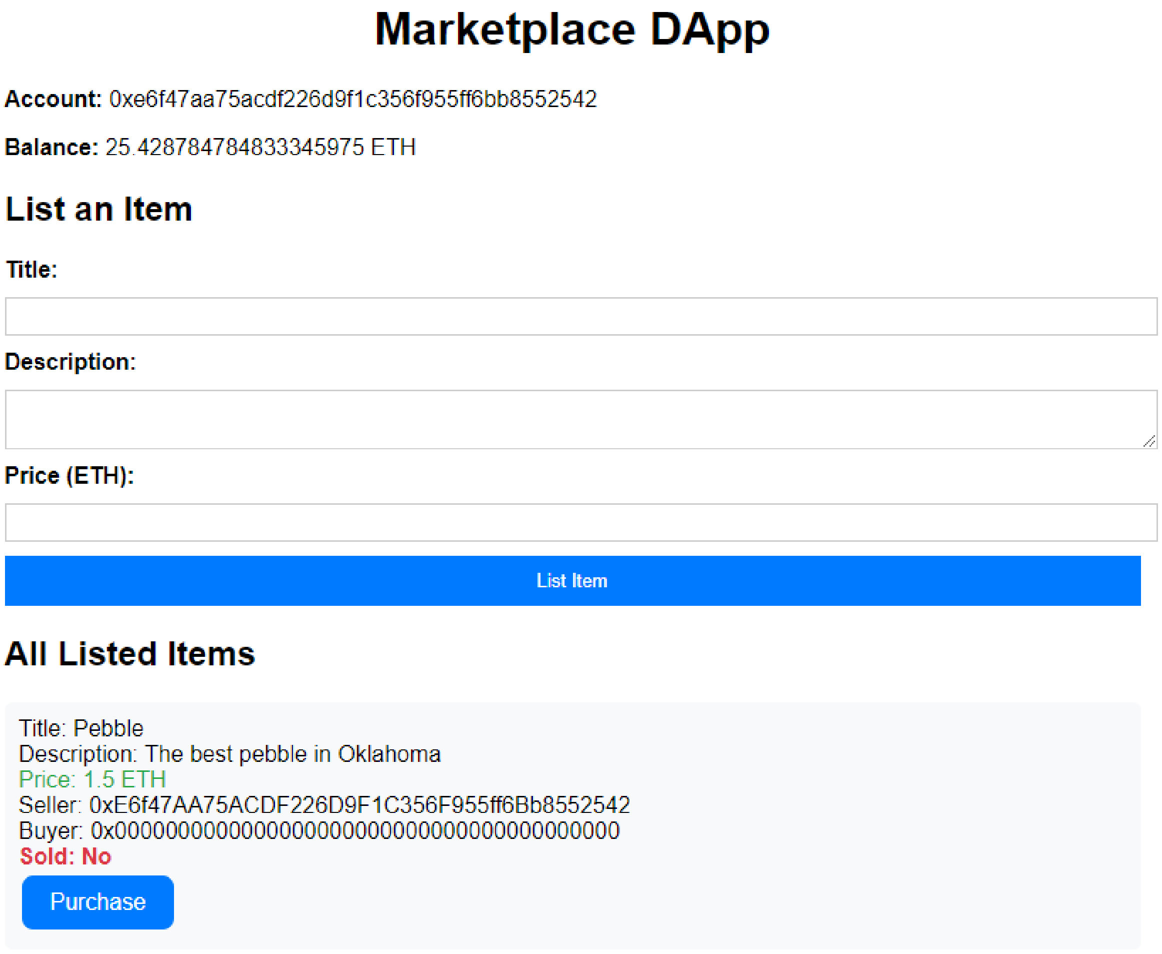
Total 0.00108077
0.00108077 SepoliaETH
Amount + gas fee Max amount: 0.00186788 SepoliaETH

After clicking confirm:



```
Item listed successfully:                                                                    app.js:325
▸ {blockHash: '0xdfc696e379d77fe746ea875f0397a72d2e3f1a066f9f4cd9ad309e12c19d3d8c', blockNumber: 5806
  453, contractAddress: null, cumulativeGasUsed: 21582036, effectiveGasPrice: '0xed32f842', …}
```

The transaction went through and now the item is listed.



Marketplace DApp

Account: 0xe6f47aa75acdf226d9f1c356f955ff6bb8552542

Balance: 25.428784784833345975 ETH

List an Item

Title:

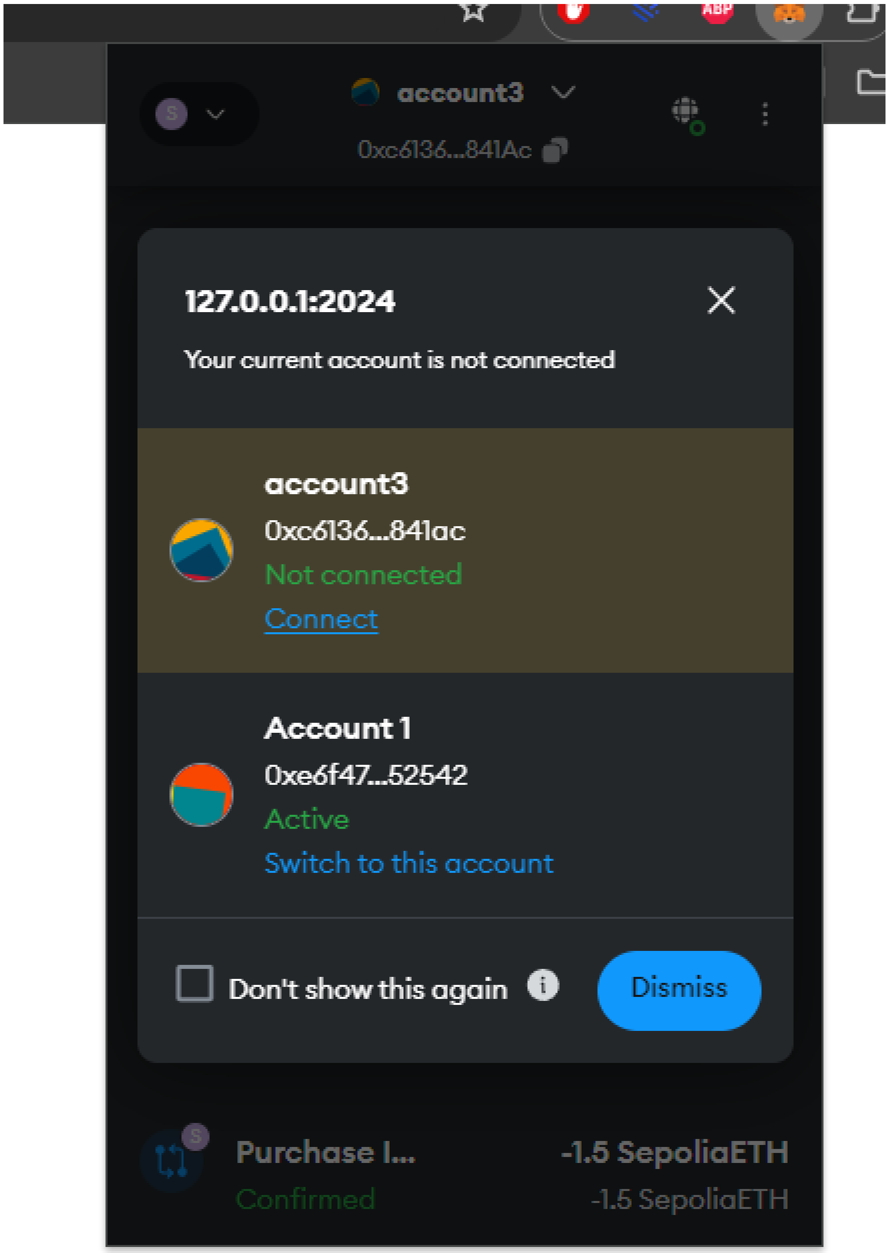
Description:

Price (ETH):

All Listed Items

Title: Pebble
Description: The best pebble in Oklahoma
Price: 1.5 ETH
Seller: 0xE6f47AA75ACDF226D9F1C356F955ff6Bb8552542
Buyer: 0x0000000000000000000000000000000000000000
Sold: No

Say if I logged into another account wallet and wanted to buy the item.



127.0.0.1:2024

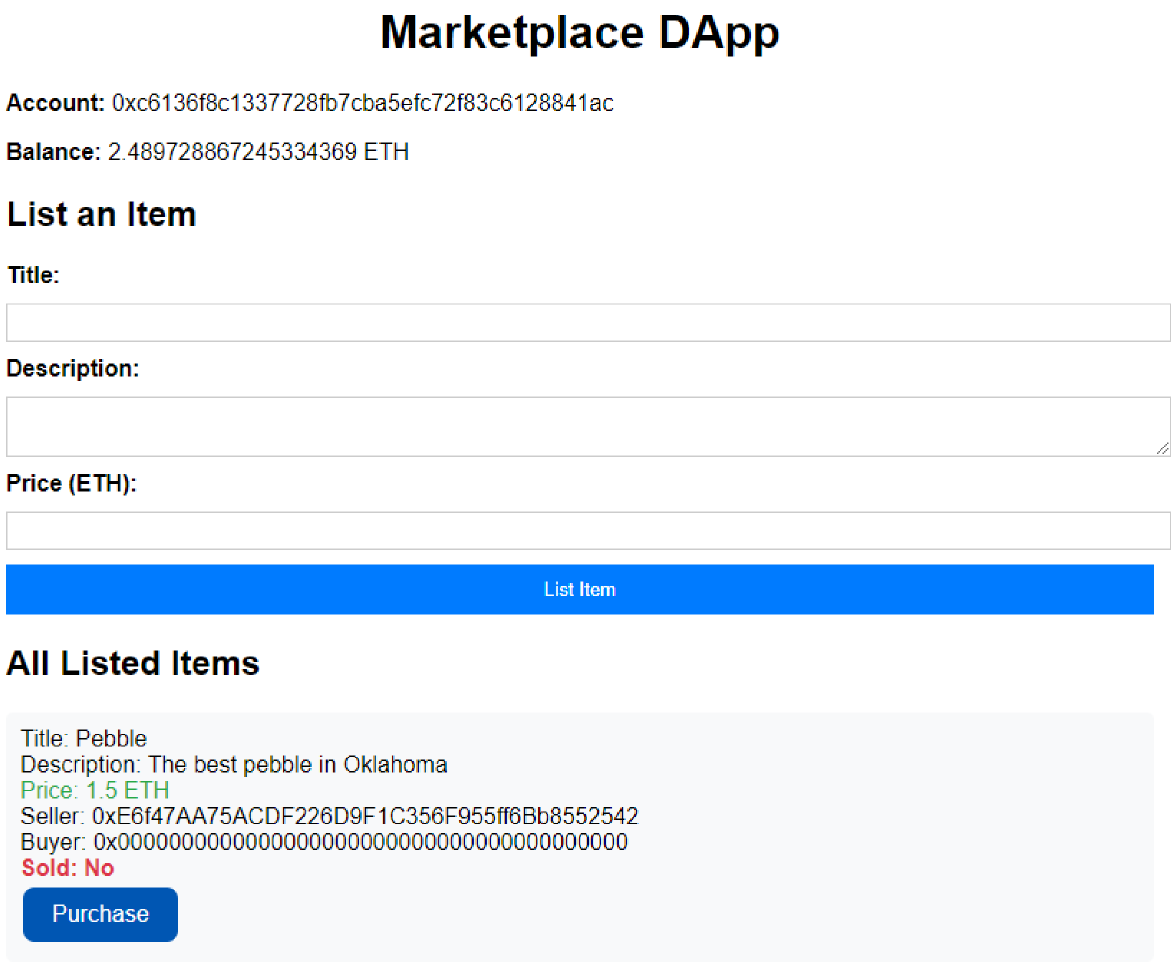
Your current account is not connected

account3
0xc6136...841ac
Not connected
Connect

Account 1
0xe6f47...52542
Active
Switch to this account

☐ Don't show this again

The new user will now see this listed item that they can buy.



Marketplace DApp

Account: 0xc6136f8c1337728fb7cba5efc72f83c6128841ac

Balance: 2.489728867245334369 ETH

List an Item

Title:

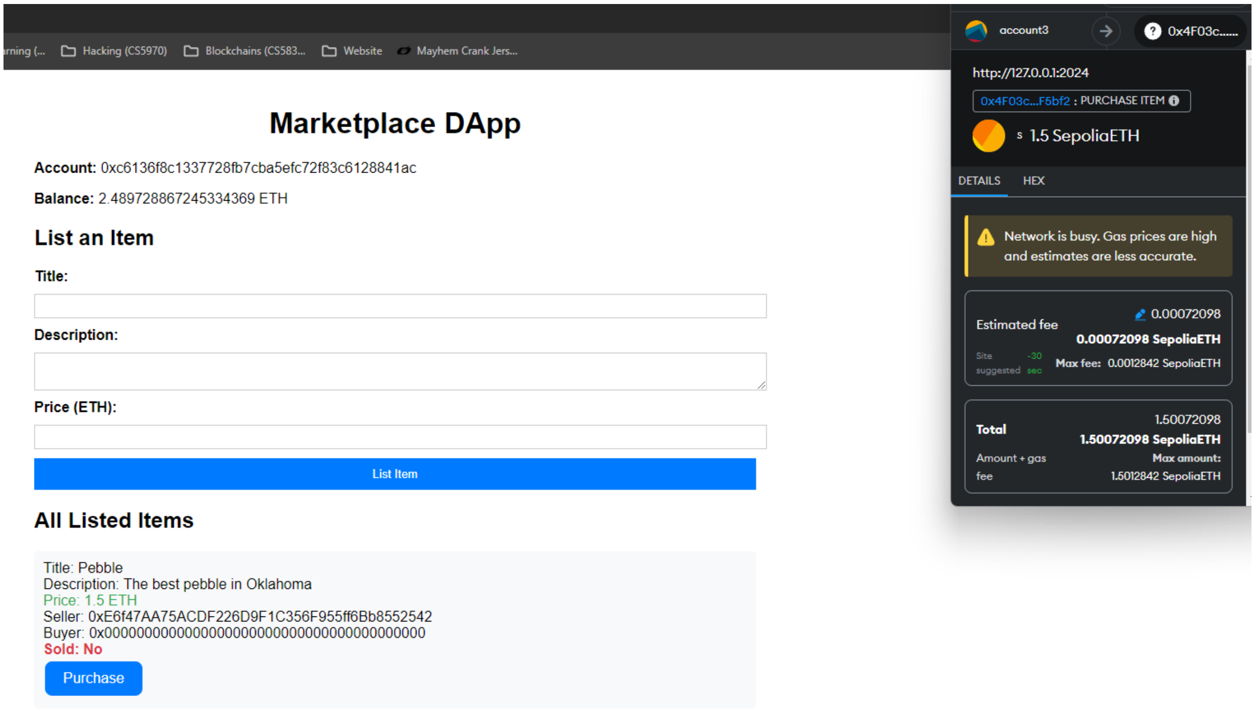
Description:

Price (ETH):

All Listed Items

Title: Pebble
Description: The best pebble in Oklahoma
Price: 1.5 ETH
Seller: 0xE6f47AA75ACDF226D9F1C356F955ff6Bb8552542
Buyer: 0x0000000000000000000000000000000000000000
Sold: No

Let’s have this user buy this item. After clicking purchase, it will prompt a transaction fee.



Marketplace DApp

Account: 0xc6136f8c1337728fb7cba5efc72f83c6128841ac

Balance: 2.489728867245334369 ETH

List an Item

Title:

Description:

Price (ETH):

All Listed Items

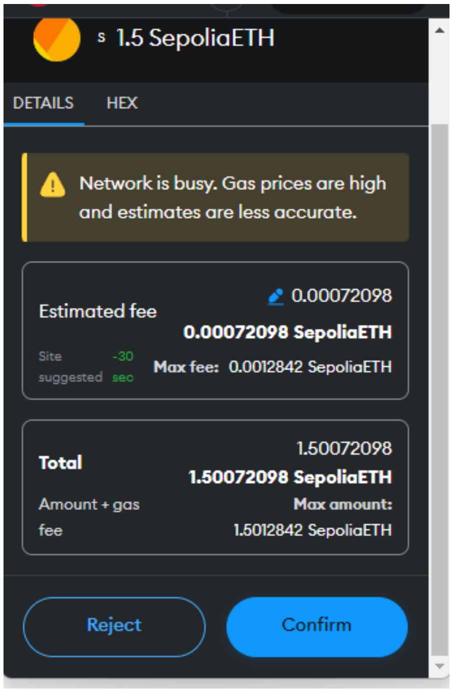
Title: Pebble
Description: The best pebble in Oklahoma
Price: 1.5 ETH
Seller: 0xE6f47AA75ACDF226D9F1C356F955ff6Bb8552542
Buyer: 0x0000000000000000000000000000000000000000
Sold: No

http://127.0.0.1:2024
0x4F03c...F5bf2 : PURCHASE ITEM
1.5 SepoliaETH

Network is busy. Gas prices are high and estimates are less accurate.

Estimated fee 0.00072098
0.00072098 SepoliaETH
Site suggested -30 sec
Max fee: 0.0012842 SepoliaETH

Total 1.50072098
1.50072098 SepoliaETH
Amount + gas fee
Max amount: 1.5012842 SepoliaETH



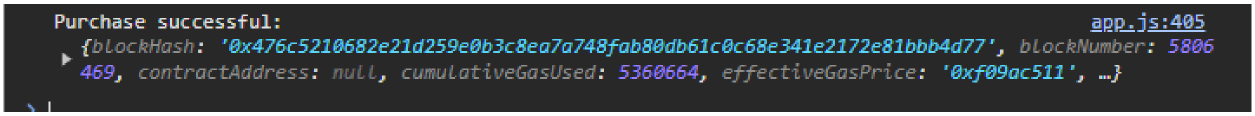
1.5 SepoliaETH

Network is busy. Gas prices are high and estimates are less accurate.

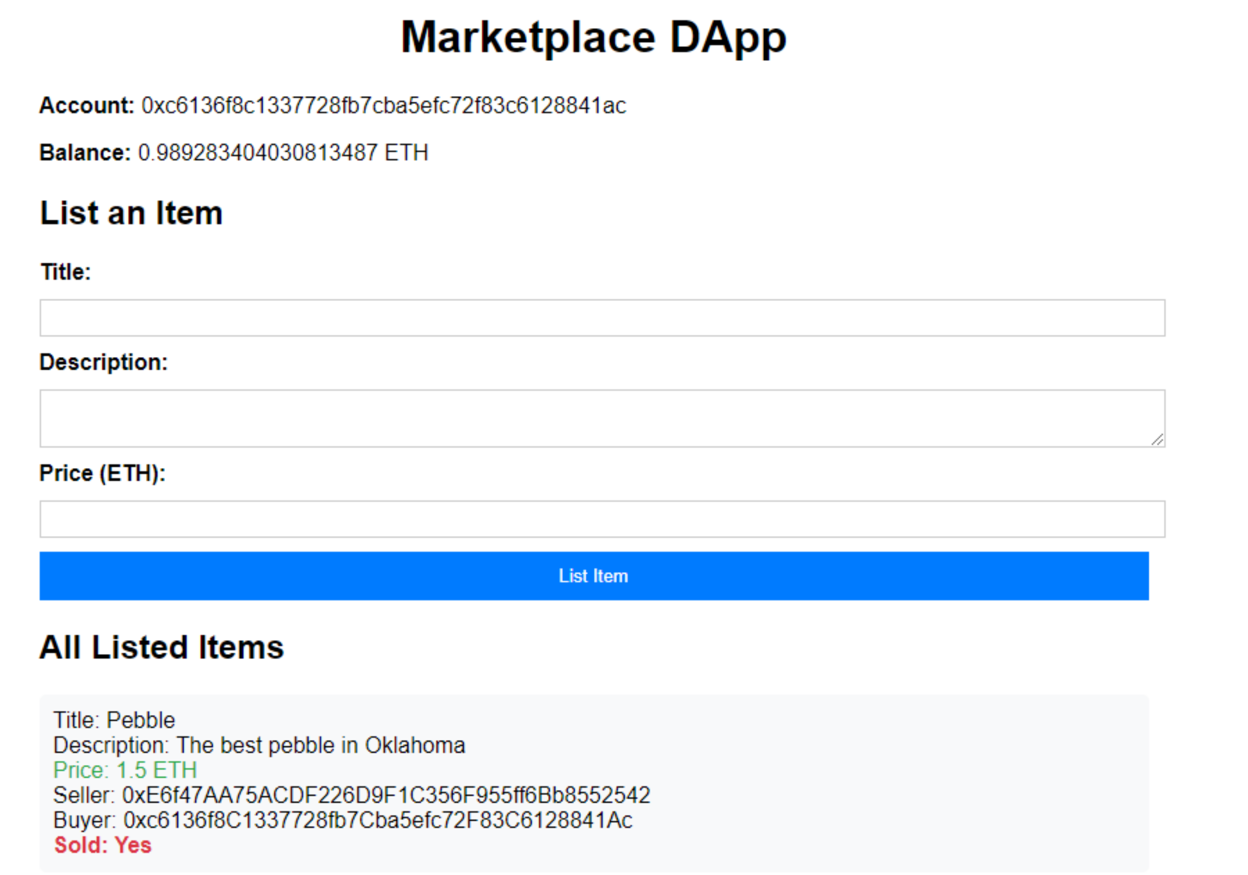
Estimated fee 0.00072098
0.00072098 SepoliaETH
Site suggested -30 sec
Max fee: 0.0012842 SepoliaETH

Total 1.50072098
1.50072098 SepoliaETH
Amount + gas fee
Max amount: 1.5012842 SepoliaETH

After clicking confirm, the transaction went through.



```
Purchase successful:                                                              app.js:405
▸ {blockHash: '0x476c5210682e21d259e0b3c8ea7a748fab80db61c0c68e341e2172e81bbb4d77', blockNumber: 5806
  469, contractAddress: null, cumulativeGasUsed: 5360664, effectiveGasPrice: '0xf09ac511', …}
```



Marketplace DApp

Account: 0xc6136f8c1337728fb7cba5efc72f83c6128841ac

Balance: 0.989283404030813487 ETH

List an Item

Title:

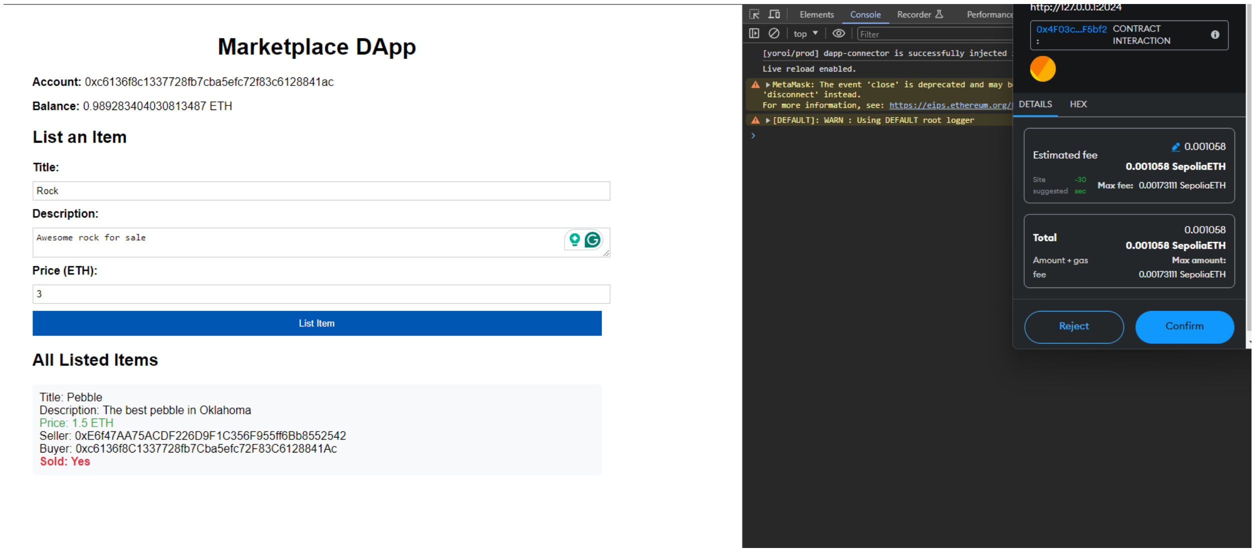
Description:

Price (ETH):

All Listed Items

Title: Pebble
Description: The best pebble in Oklahoma
Price: 1.5 ETH
Seller: 0xE6f47AA75ACDF226D9F1C356F955ff6Bb8552542
Buyer: 0xc6136f8C1337728fb7Cba5efc72F83C6128841Ac
Sold: Yes

Say this user wanted to list an item of their own.



```
[yoroi/prod] dapp-connector is successfully injected
Live reload enabled.
▸MetaMask: The event 'close' is deprecated and may b
'disconnect' instead.
For more information, see: https://eips.ethereum.org/
▸[DEFAULT]: WARN : Using DEFAULT root logger
```

http://127.0.0.1:2024

0x4F03c...F5bf2 : CONTRACT INTERACTION

Estimated fee 0.001058
0.001058 SepoliaETH
Site suggested -30 sec Max fee: 0.00173111 SepoliaETH

Total 0.001058
0.001058 SepoliaETH
Amount + gas fee Max amount: 0.00173111 SepoliaETH

Marketplace DApp

Account: 0xc6136f8c1337728fb7cba5efc72f83c6128841ac

Balance: 0.989283404030813487 ETH

List an Item

Title:

Rock

Description:

Awesome rock for sale

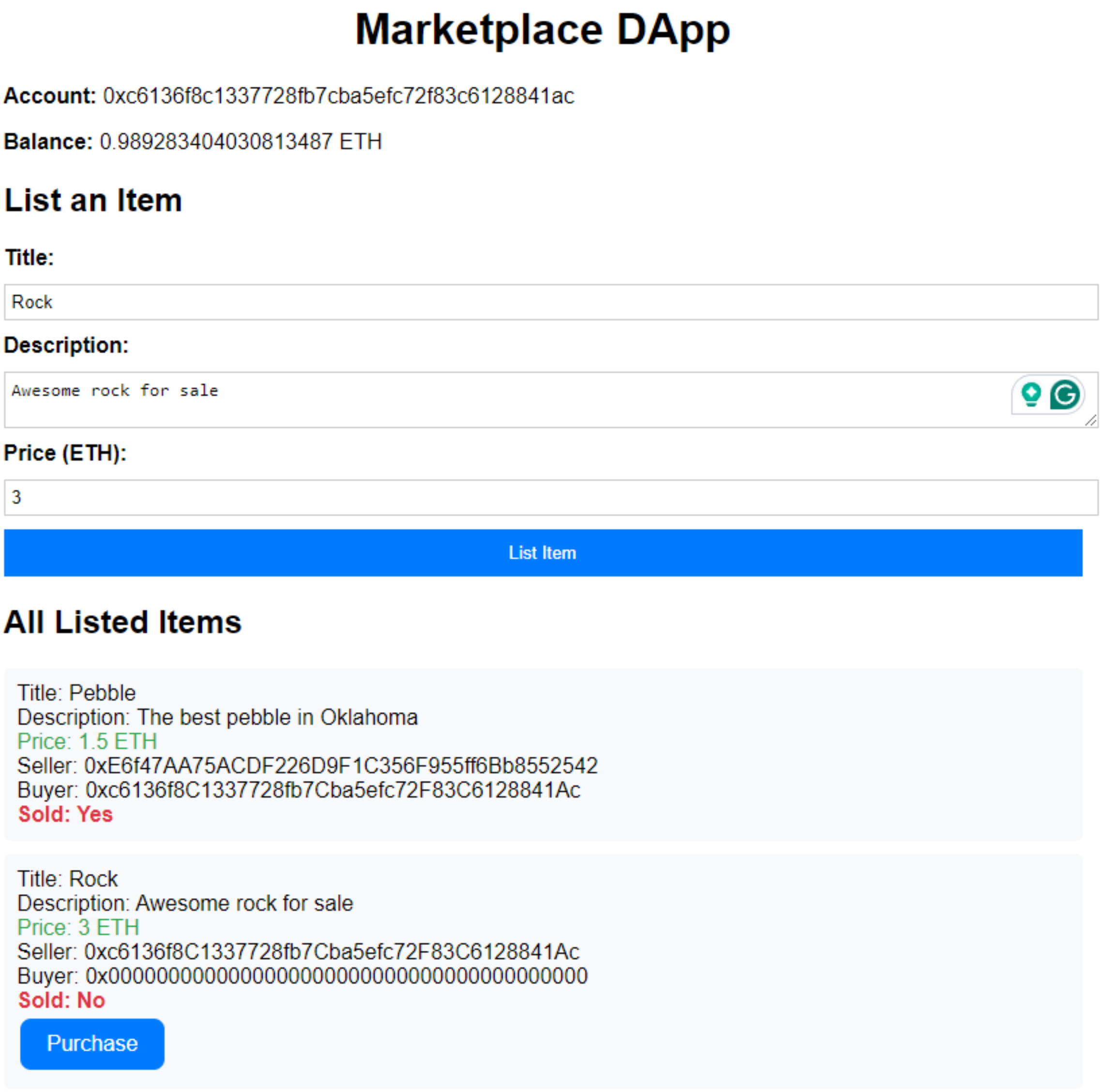
Price (ETH):

3

All Listed Items

Title: Pebble
Description: The best pebble in Oklahoma
Price: 1.5 ETH
Seller: 0xE6f47AA75ACDF226D9F1C356F955ff6Bb8552542
Buyer: 0xc6136f8C1337728fb7Cba5efc72F83C6128841Ac
Sold: Yes

After paying the transaction fee, their listed item will be on the marketplace.



Marketplace DApp

Account: 0xc6136f8c1337728fb7cba5efc72f83c6128841ac

Balance: 0.989283404030813487 ETH

List an Item

Title:

Rock

Description:

Awesome rock for sale

Price (ETH):

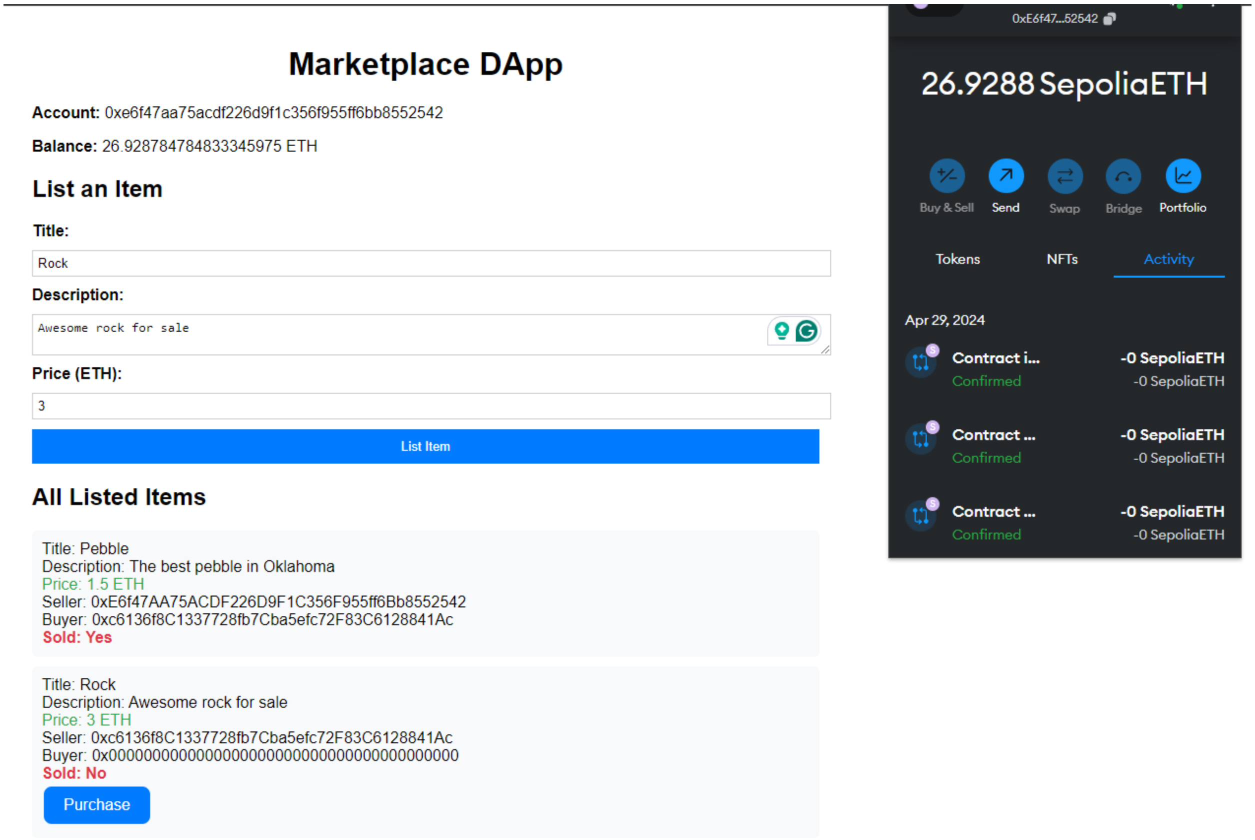
3

All Listed Items

Title: Pebble
Description: The best pebble in Oklahoma
Price: 1.5 ETH
Seller: 0xE6f47AA75ACDF226D9F1C356F955ff6Bb8552542
Buyer: 0xc6136f8C1337728fb7Cba5efc72F83C6128841Ac
Sold: Yes

Title: Rock
Description: Awesome rock for sale
Price: 3 ETH
Seller: 0xc6136f8C1337728fb7Cba5efc72F83C6128841Ac
Buyer: 0x0000000000000000000000000000000000000000
Sold: No

Let’s log into another account that wanted to buy that item.



0xE6f47...52542

26.9288 SepoliaETH

Apr 29, 2024

Contract i... Confirmed -0 SepoliaETH -0 SepoliaETH

Contract ... Confirmed -0 SepoliaETH -0 SepoliaETH

Contract ... Confirmed -0 SepoliaETH -0 SepoliaETH

Marketplace DApp

Account: 0xe6f47aa75acdf226d9f1c356f955ff6bb8552542

Balance: 26.928784784833345975 ETH

List an Item

Title:

Rock

Description:

Awesome rock for sale

Price (ETH):

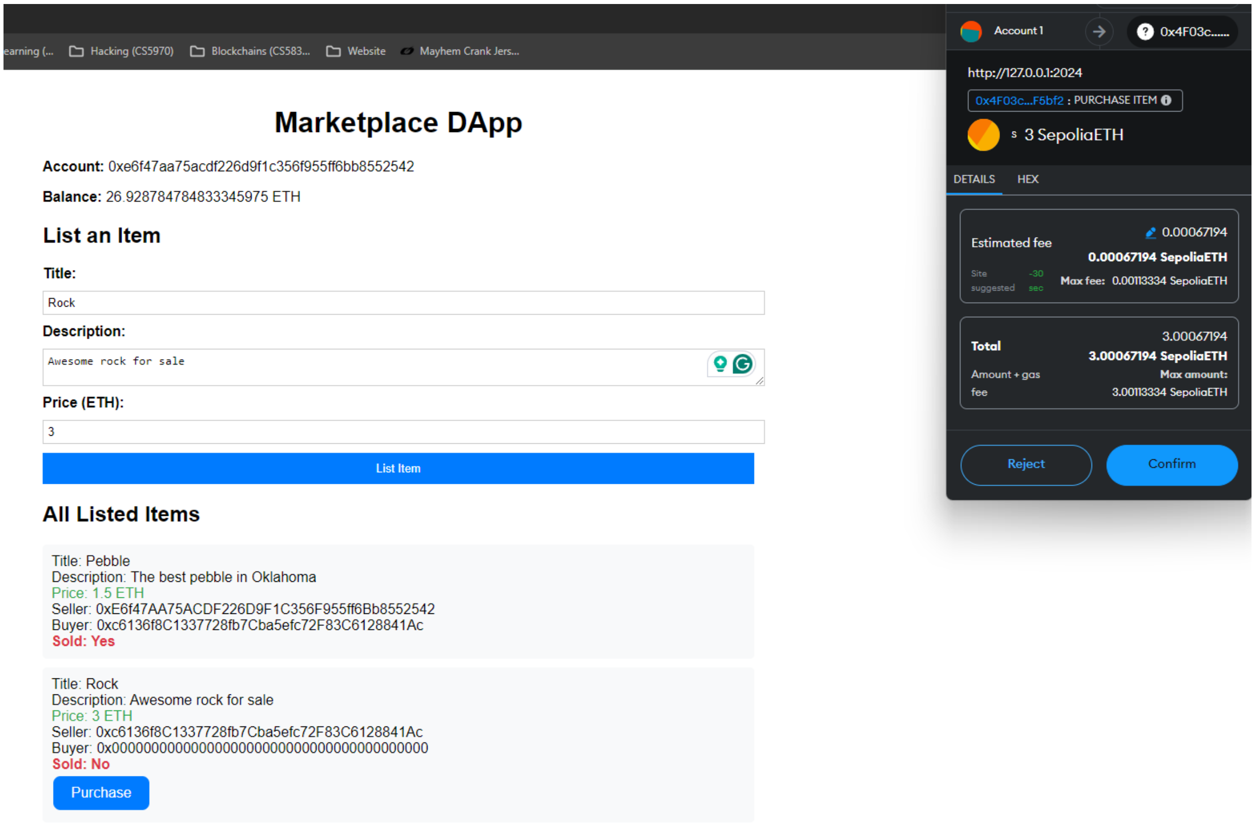
3

All Listed Items

Title: Pebble
Description: The best pebble in Oklahoma
Price: 1.5 ETH
Seller: 0xE6f47AA75ACDF226D9F1C356F955ff6Bb8552542
Buyer: 0xc6136f8C1337728fb7Cba5efc72F83C6128841Ac
Sold: Yes

Title: Rock
Description: Awesome rock for sale
Price: 3 ETH
Seller: 0xc6136f8C1337728fb7Cba5efc72F83C6128841Ac
Buyer: 0x0000000000000000000000000000000000000000
Sold: No

The user will see what items have already been sold and what items are available to buy. After this user purchases the item, the listed item is updated to sold.



Marketplace DApp

Account: 0xe6f47aa75acdf226d9f1c356f955ff6bb8552542

Balance: 26.928784784833345975 ETH

List an Item

Title:

Rock

Description:

Awesome rock for sale

Price (ETH):

3

List Item

All Listed Items

Title: Pebble
Description: The best pebble in Oklahoma
Price: 1.5 ETH
Seller: 0xE6f47AA75ACDF226D9F1C356F955ff6Bb8552542
Buyer: 0xc6136f8C1337728fb7Cba5efc72F83C6128841Ac
Sold: Yes

Title: Rock
Description: Awesome rock for sale
Price: 3 ETH
Seller: 0xc6136f8C1337728fb7Cba5efc72F83C6128841Ac
Buyer: 0x0000000000000000000000000000000000000000
Sold: No

Purchase

Account 1
0x4F03c......

http://127.0.0.1:2024

0x4F03c...F5bf2 : PURCHASE ITEM

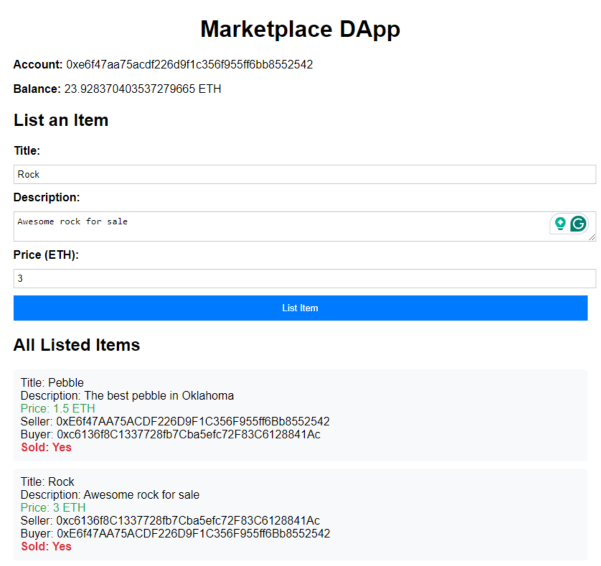
3 SepoliaETH

DETAILS HEX

Estimated fee 0.00067194
0.00067194 SepoliaETH
Site suggested -30 sec
Max fee: 0.00113334 SepoliaETH

Total 3.00067194
3.00067194 SepoliaETH
Amount + gas fee
Max amount: 3.00113334 SepoliaETH

Reject
Confirm



Marketplace DApp

Account: 0xe6f47aa75acdf226d9f1c356f955ff6bb8552542

Balance: 23.928370403537279665 ETH

List an Item

Title:

Rock

Description:

Awesome rock for sale

Price (ETH):

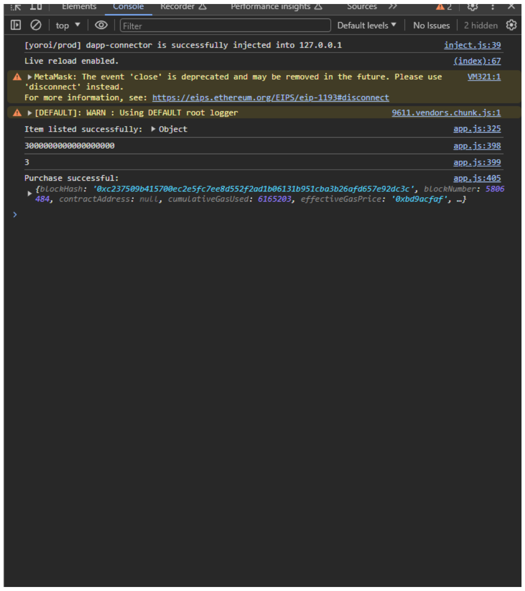
3

List Item

All Listed Items

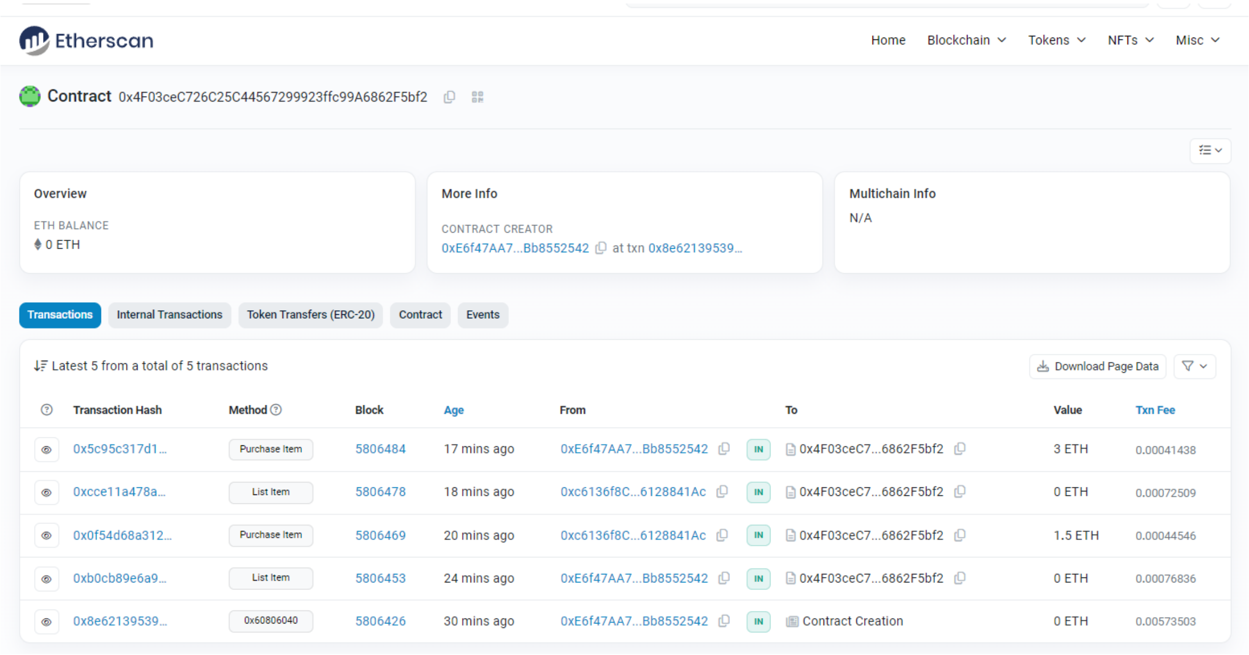
Title: Pebble
Description: The best pebble in Oklahoma
Price: 1.5 ETH
Seller: 0xE6f47AA75ACDF226D9F1C356F955ff6Bb8552542
Buyer: 0xc6136f8C1337728fb7Cba5efc72F83C6128841Ac
Sold: Yes

Title: Rock
Description: Awesome rock for sale
Price: 3 ETH
Seller: 0xc6136f8C1337728fb7Cba5efc72F83C6128841Ac
Buyer: 0xE6f47AA75ACDF226D9F1C356F955ff6Bb8552542
Sold: Yes



```
[yoroi/prod] dapp-connector is successfully injected into 127.0.0.1          inject.js:39
Live reload enabled.                                                         (index):67
▶MetaMask: The event 'close' is deprecated and may be removed in the future. Please use    VM321:1
'disconnect' instead.
For more information, see: https://eips.ethereum.org/EIPS/eip-1193#disconnect
▶[DEFAULT]: WARN : Using DEFAULT root logger                         9611.vendors.chunk.js:1
Item listed successfully: ▶ Object                                           app.js:325
3000000000000000000                                                          app.js:398
3                                                                            app.js:399
Purchase successful:                                                         app.js:405
▶ {blockHash: '0xc237509b415700ec2e5fc7ee8d552f2ad1b06131b951cba3b26afd657e92dc3c', blockNumber: 5806
  484, contractAddress: null, cumulativeGasUsed: 6165203, effectiveGasPrice: '0xbd9acfaf', …}
>
```

Transaction confirmations on the Marketplace.sol contract:



Contract 0x4F03ceC726C25C44567299923ffc99A6862F5bf2

Overview

ETH BALANCE
0 ETH

More Info

CONTRACT CREATOR
0xE6f47AA7...Bb8552542 at txn 0x8e62139539...

Multichain Info
N/A

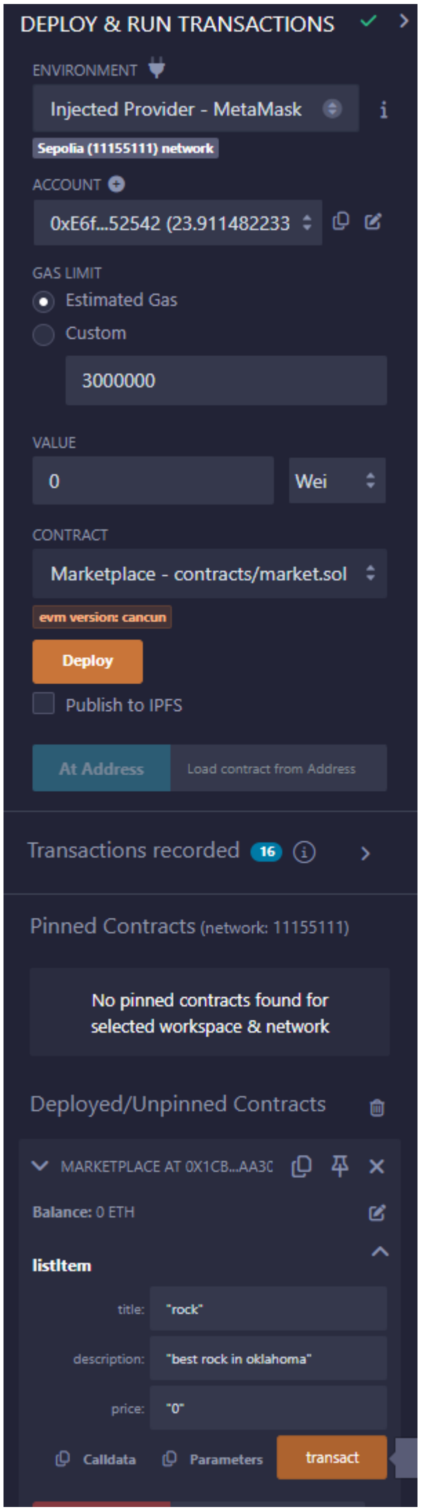
Latest 5 from a total of 5 transactions

Transaction Hash	Method	Block	Age	From		To	Value	Txn Fee
0x5c95c317d1...	Purchase Item	5806484	17 mins ago	0xE6f47AA7...Bb8552542	IN	0x4F03ceC7...6862F5bf2	3 ETH	0.00041438
0xcce11a478a...	List Item	5806478	18 mins ago	0xc6136f8C...6128841Ac	IN	0x4F03ceC7...6862F5bf2	0 ETH	0.00072509
0x0f54d68a312...	Purchase Item	5806469	20 mins ago	0xc6136f8C...6128841Ac	IN	0x4F03ceC7...6862F5bf2	1.5 ETH	0.00044546
0xb0cb89e6a9...	List Item	5806453	24 mins ago	0xE6f47AA7...Bb8552542	IN	0x4F03ceC7...6862F5bf2	0 ETH	0.00076836
0x8e62139539...	0x60806040	5806426	30 mins ago	0xE6f47AA7...Bb8552542	IN	Contract Creation	0 ETH	0.00573503

Testing:

Test 1: Seller tries to sell item with 0 Wei

Account 1 attempts to list an item for 0 Wei or 0 ETH



DEPLOY & RUN TRANSACTIONS

ENVIRONMENT

Injected Provider - MetaMask

Sepolia (11155111) network

ACCOUNT

0xE6f...52542 (23.911482233

GAS LIMIT

● Estimated Gas

○ Custom

3000000

VALUE

0 Wei

CONTRACT

Marketplace - contracts/market.sol

evm version: cancun

Deploy

□ Publish to IPFS

At Address | Load contract from Address

Transactions recorded 16

Pinned Contracts (network: 11155111)

No pinned contracts found for selected workspace & network

Deployed/Unpinned Contracts

MARKETPLACE AT 0X1CB...AA3C

Balance: 0 ETH

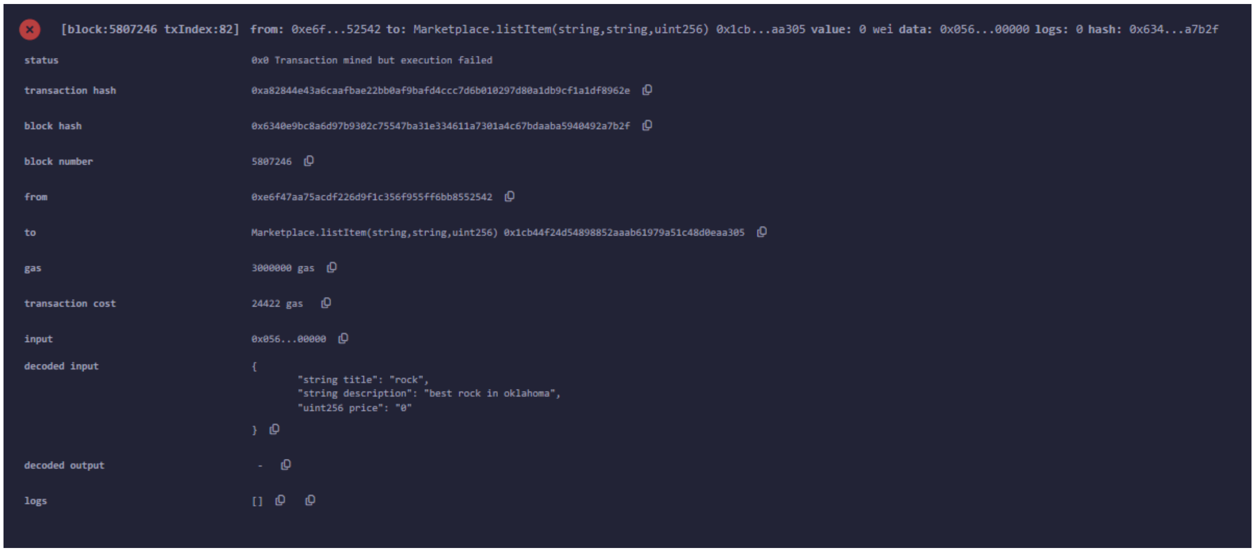
listItem

title: "rock"

description: "best rock in oklahoma"

price: "0"

Calldata | Parameters | transact

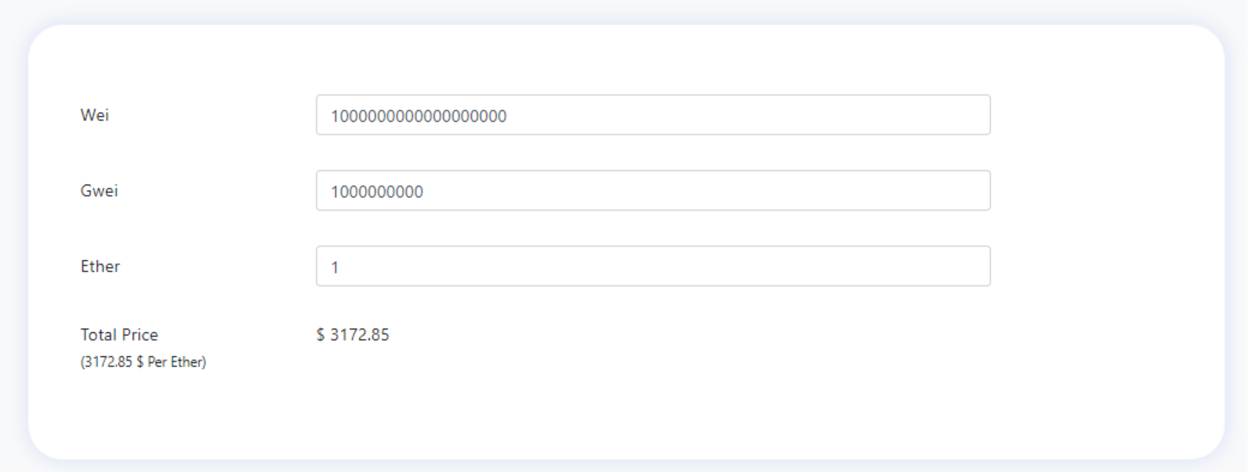


```
[block:5807246 txIndex:82] from: 0xe6f...52542 to: Marketplace.listItem(string,string,uint256) 0x1cb...aa305 value: 0 wei data: 0x056...00000 logs: 0 hash: 0x634...a7b2f
status                 0x0 Transaction mined but execution failed
transaction hash       0xa82844e43a6caafbae22bb0af9bafd4ccc7d6b010297d80a1db9cf1a1df8962e
block hash             0x6340e9bc8a6d97b9302c75547ba31e334611a7301a4c67bdaaba5940492a7b2f
block number           5807246
from                   0xe6f47aa75acdf226d9f1c356f955ff6bb8552542
to                     Marketplace.listItem(string,string,uint256) 0x1cb44f24d54898852aaab61979a51c48d0eaa305
gas                    3000000 gas
transaction cost       24422 gas
input                  0x056...00000
decoded input          {
                               "string title": "rock",
                               "string description": "best rock in oklahoma",
                               "uint256 price": "0"
                       }
decoded output          -
logs                   []
```

This listItem transaction fails because the amount needs to be at least 1 WEI. TEST PASSED!

Test 2: Seller sells item with over 1 Wei

Account 1 attempts to list an item for 1000000000000000000 Wei or 1 ETH

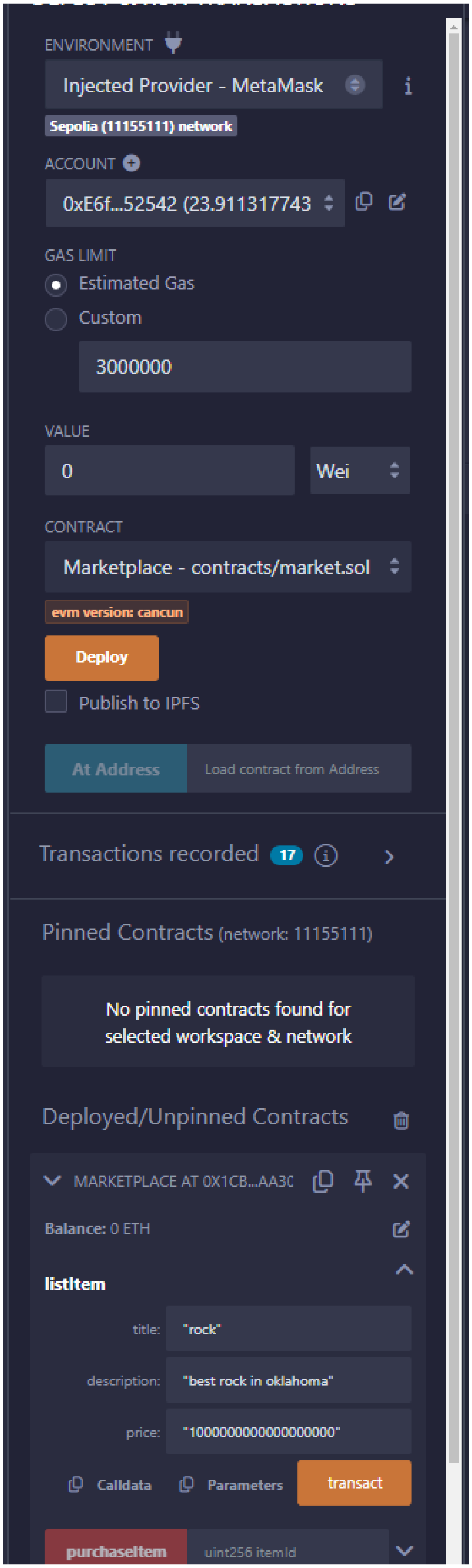


Wei 1000000000000000000

Gwei 1000000000

Ether 1

Total Price $ 3172.85
(3172.85 $ Per Ether)



ENVIRONMENT

Injected Provider - MetaMask

Sepolia (11155111) network

ACCOUNT

0xE6f...52542 (23.911317743

GAS LIMIT

● Estimated Gas

○ Custom

3000000

VALUE

0 Wei

CONTRACT

Marketplace - contracts/market.sol

evm version: cancun

□ Publish to IPFS

Load contract from Address

Transactions recorded 17

Pinned Contracts (network: 11155111)

No pinned contracts found for selected workspace & network

Deployed/Unpinned Contracts

MARKETPLACE AT 0X1CB...AA30

Balance: 0 ETH

listItem

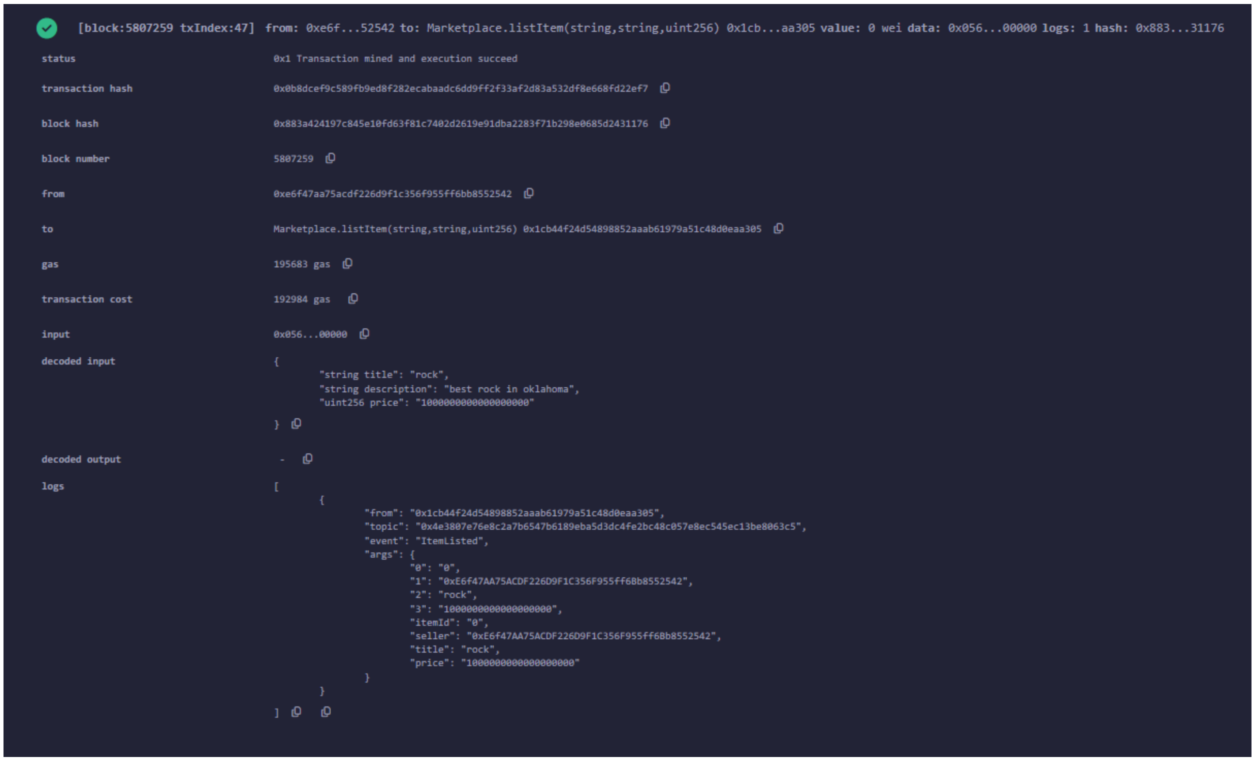
title: "rock"

description: "best rock in oklahoma"

price: "1000000000000000000"

uint256 itemId

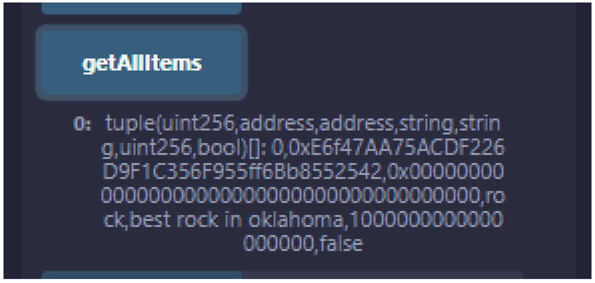
listItem transaction goes through because it is at least 1 Wei.



```
[block:5807259 txIndex:47] from: 0xe6f...52542 to: Marketplace.listItem(string,string,uint256) 0x1cb...aa305 value: 0 wei data: 0x056...00000 logs: 1 hash: 0x883...31176
status                    0x1 Transaction mined and execution succeed
transaction hash          0x0b8dcef9c589fb9ed8f282ecabaadc6dd9ff2f33af2d83a532df8e668fd22ef7
block hash                0x883a424197c845e10fd63f81c7402d2619e91dba2283f71b298e0685d2431176
block number              5807259
from                      0xe6f47aa75acdf226d9f1c356f955ff6bb8552542
to                        Marketplace.listItem(string,string,uint256) 0x1cb44f24d54898852aaab61979a51c48d0eaa305
gas                       195683 gas
transaction cost          192984 gas
input                     0x056...00000
decoded input             {
                                "string title": "rock",
                                "string description": "best rock in oklahoma",
                                "uint256 price": "1000000000000000000"
                          }
decoded output             -
logs                      [
                                {
                                        "from": "0x1cb44f24d54898852aaab61979a51c48d0eaa305",
                                        "topic": "0x4e3807e76e8c2a7b6547b6189eba5d3dc4fe2bc48c057e8ec545ec13be8063c5",
                                        "event": "ItemListed",
                                        "args": {
                                                "0": "0",
                                                "1": "0xE6f47AA75ACDF226D9F1C356F955ff6Bb8552542",
                                                "2": "rock",
                                                "3": "1000000000000000000",
                                                "itemId": "0",
                                                "seller": "0xE6f47AA75ACDF226D9F1C356F955ff6Bb8552542",
                                                "title": "rock",
                                                "price": "1000000000000000000"
                                        }
                                }
                          ]
```

TEST PASSED!

Call getAllItems to confirm listing:

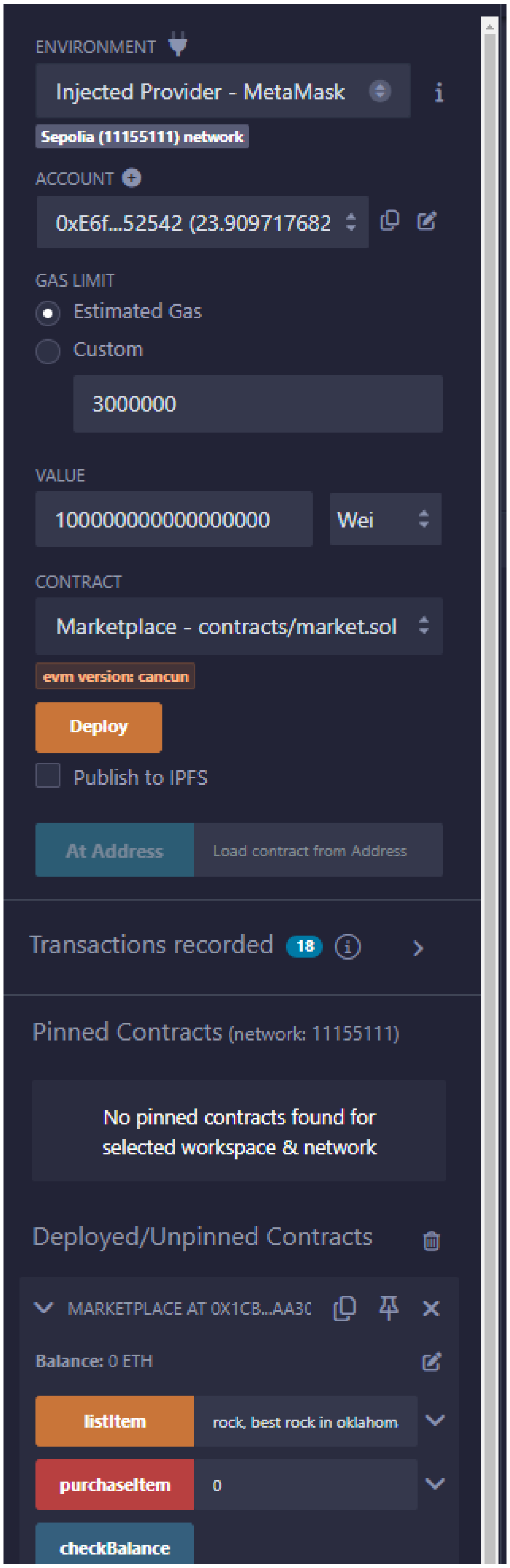


```
0: tuple(uint256,address,address,string,string,uint256,bool)[]: 0,0xE6f47AA75ACDF226D9F1C356F955ff6Bb8552542,0x0000000000000000000000000000000000000000,rock,best rock in oklahoma,1000000000000000000,false
```

TRUE

Test 3: Buyer tries to buy their own listed items.

Account 1 tries to buy their own listing. (itemId = 0)



ENVIRONMENT

Injected Provider - MetaMask

Sepolia (11155111) network

ACCOUNT

0xE6f...52542 (23.909717682

GAS LIMIT

● Estimated Gas

○ Custom

3000000

VALUE

1000000000000000000 Wei

CONTRACT

Marketplace - contracts/market.sol

evm version: cancun

□ Publish to IPFS

Transactions recorded 18

Pinned Contracts (network: 11155111)

No pinned contracts found for selected workspace & network

Deployed/Unpinned Contracts

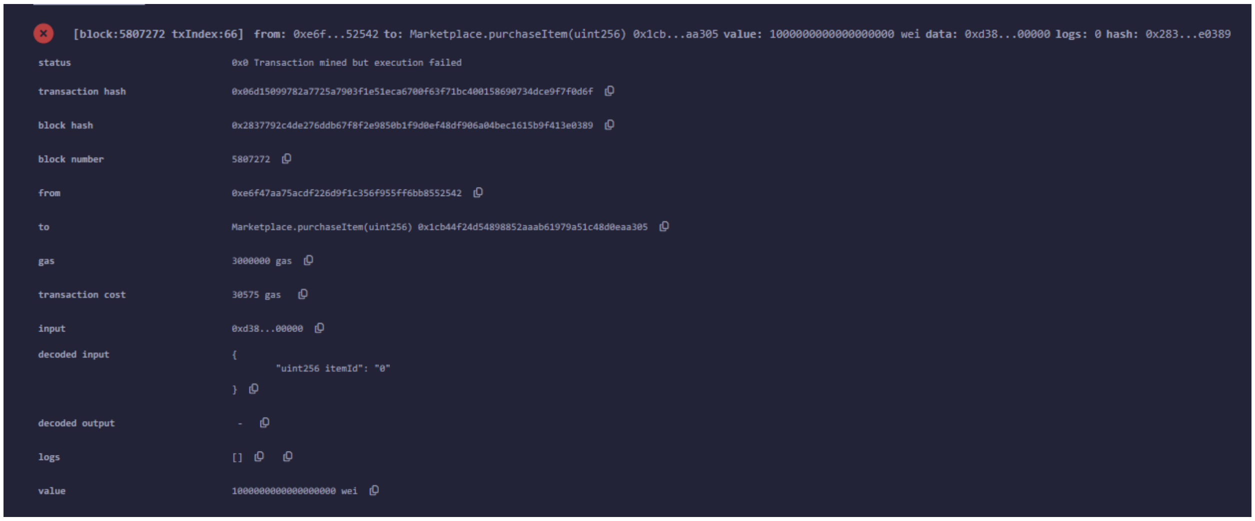
MARKETPLACE AT 0X1CB...AA3C

Balance: 0 ETH

listItem: rock, best rock in oklahom

purchaseItem: 0

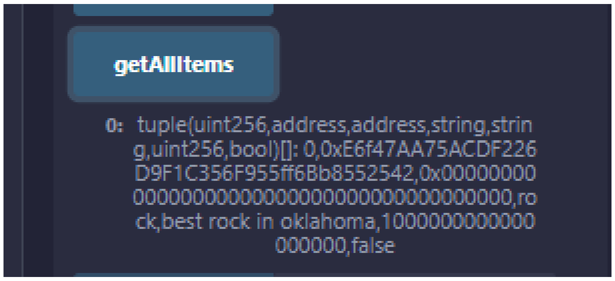
The purchaseItem transaction fails because in the contract, we specified that the buyer cannot be owner of the listed item:



```
[block:5807272 txIndex:66] from: 0xe6f...52542 to: Marketplace.purchaseItem(uint256) 0x1cb...aa305 value: 1000000000000000000 wei data: 0xd38...00000 logs: 0 hash: 0x283...e0389
status                  0x0 Transaction mined but execution failed
transaction hash        0x06d15099782a7725a7903f1e51eca6700f63f71bc400158690734dce9f7f0d6f
block hash              0x2837792c4de276ddb67f8f2e9850b1f9d0ef48df906a04bec1615b9f413e0389
block number            5807272
from                    0xe6f47aa75acdf226d9f1c356f955ff6bb8552542
to                      Marketplace.purchaseItem(uint256) 0x1cb44f24d54898852aaab61979a51c48d0eaa305
gas                     3000000 gas
transaction cost        30575 gas
input                   0xd38...00000
decoded input           {
                                "uint256 itemId": "0"
                        }
decoded output           -
logs                    []
value                   1000000000000000000 wei
```

TEST PASSED.

Call getAllItems to confirm that it has not been sold:

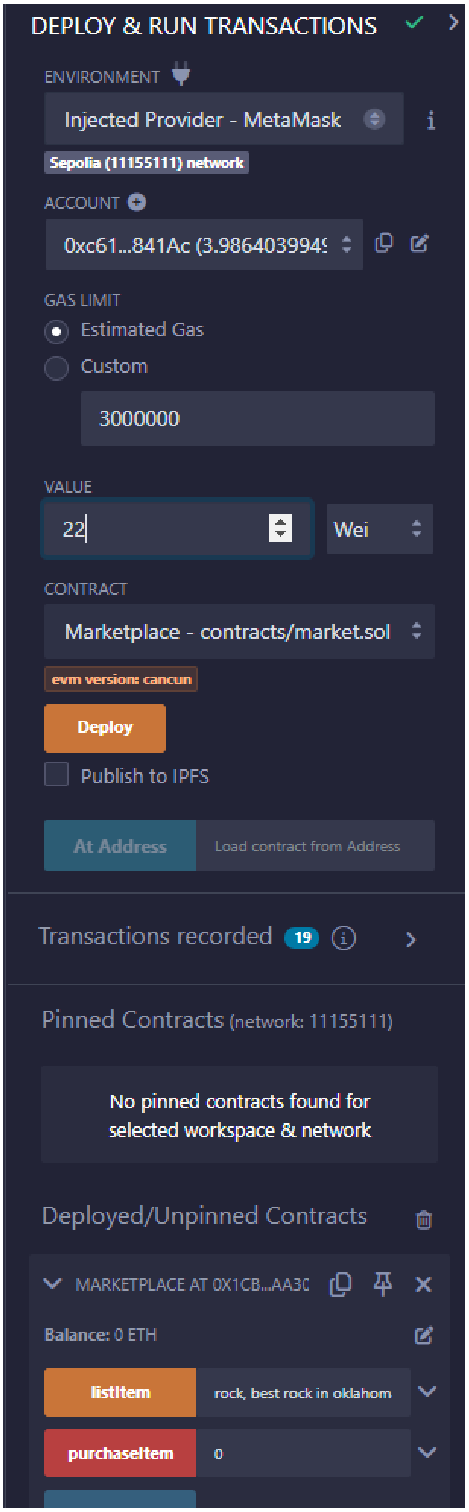


```
0: tuple(uint256,address,address,string,string,uint256,bool)[]: 0,0xE6f47AA75ACDF226D9F1C356F955ff6Bb8552542,0x0000000000000000000000000000000000000000,rock,best rock in oklahoma,1000000000000000000,false
```

TRUE.

Test 4: Buyer tries to buy item under the list items value.

Account 2 attempts to buy an item 22 Wei when the listed item was listed for 1000000000000000000 Wei or 1 ETH.



DEPLOY & RUN TRANSACTIONS

ENVIRONMENT

Injected Provider - MetaMask

Sepolia (11155111) network

ACCOUNT

0xc61...841Ac (3.98640399

GAS LIMIT

● Estimated Gas

○ Custom

3000000

VALUE

22 Wei

CONTRACT

Marketplace - contracts/market.sol

evm version: cancun

Transactions recorded 19

Pinned Contracts (network: 11155111)

No pinned contracts found for selected workspace & network

Deployed/Unpinned Contracts

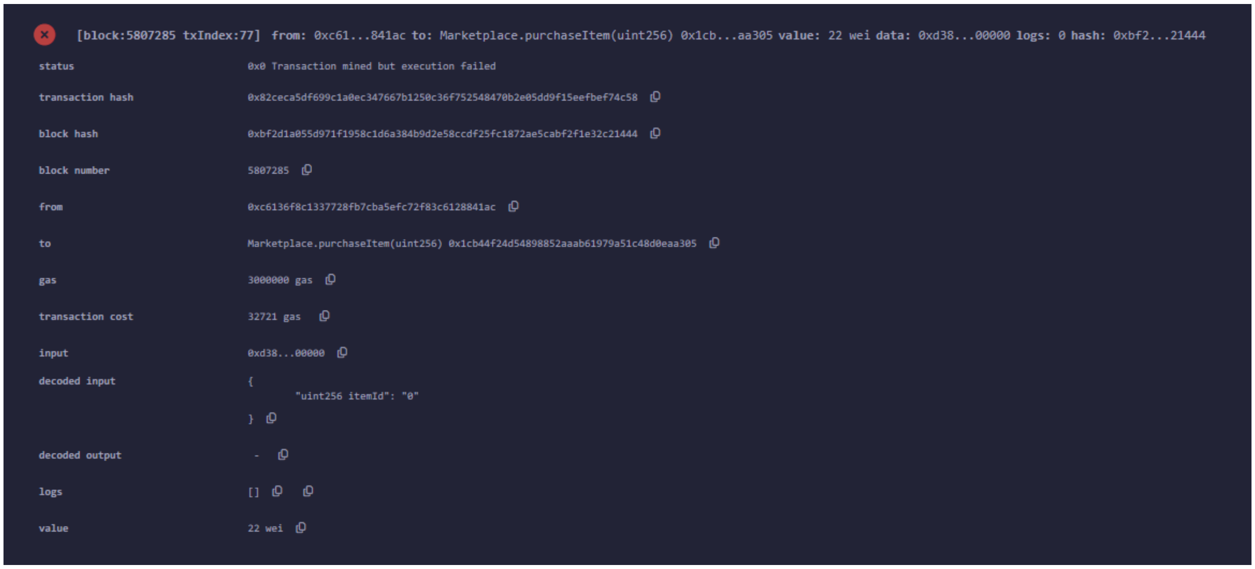
MARKETPLACE AT 0X1CB...AA3C

Balance: 0 ETH

listItem: rock, best rock in oklahom

purchaseItem: 0

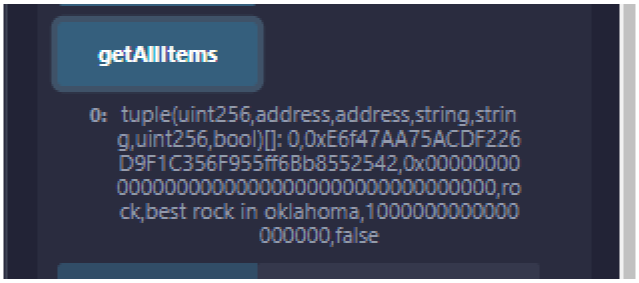
The purchaseItem transaction fails because in the contract, we specified that the buyer cannot be must pay the listed item price.



```
[block:5807285 txIndex:77]  from: 0xc61...841ac to: Marketplace.purchaseItem(uint256) 0x1cb...aa305 value: 22 wei data: 0xd38...00000 logs: 0 hash: 0xbf2...21444
status                      0x0 Transaction mined but execution failed
transaction hash            0x82ceca5df699c1a0ec347667b1250c36f752548470b2e05dd9f15eefbef74c58
block hash                  0xbf2d1a055d971f1958c1d6a384b9d2e58ccdf25fc1872ae5cabf2f1e32c21444
block number                5807285
from                        0xc6136f8c1337728fb7cba5efc72f83c6128841ac
to                          Marketplace.purchaseItem(uint256) 0x1cb44f24d54898852aaab61979a51c48d0eaa305
gas                         3000000 gas
transaction cost            32721 gas
input                       0xd38...00000
decoded input               {
                                  "uint256 itemId": "0"
                            }
decoded output               -
logs                        []
value                       22 wei
```

TEST PASSED.

Call getAllItems to confirm that it has not been sold:

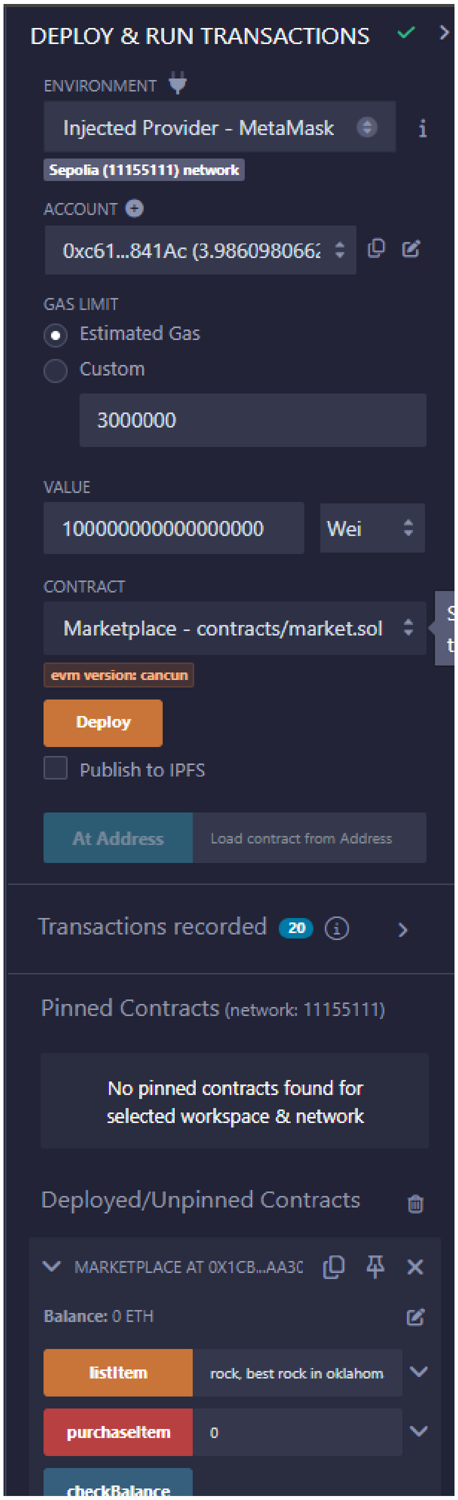


```
0: tuple(uint256,address,address,string,string,uint256,bool)[]: 0,0xE6f47AA75ACDF226D9F1C356F955ff6Bb8552542,0x0000000000000000000000000000000000000000,rock,best rock in oklahoma,1000000000000000000,false
```

TRUE!

Test 5: Buyer tries to buy the listed item equal to the listed item’s value.

Account 2 attempts to buy an item for 1000000000000000000 Wei and the listed item was listed for 1000000000000000000 Wei or 1 ETH.



DEPLOY & RUN TRANSACTIONS

ENVIRONMENT

Injected Provider - MetaMask

Sepolia (11155111) network

ACCOUNT

0xc61...841Ac (3.9860980662

GAS LIMIT

● Estimated Gas

○ Custom

3000000

VALUE

100000000000000000 Wei

CONTRACT

Marketplace - contracts/market.sol

evm version: cancun

Deploy

□ Publish to IPFS

At Address | Load contract from Address

Transactions recorded 20

Pinned Contracts (network: 11155111)

No pinned contracts found for selected workspace & network

Deployed/Unpinned Contracts

MARKETPLACE AT 0X1CB...AA3C

Balance: 0 ETH

listItem | rock, best rock in oklahom

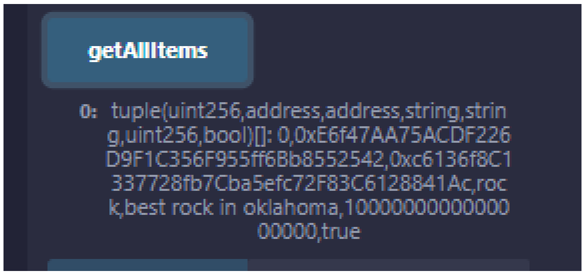
purchaseItem | 0

checkBalance

The purchaseItem transaction went through because the buyer paid the exact amount of the listed item price.

TEST PASSED.

Call getAllItems to confirm the item is sold and has the correct buyer and seller:

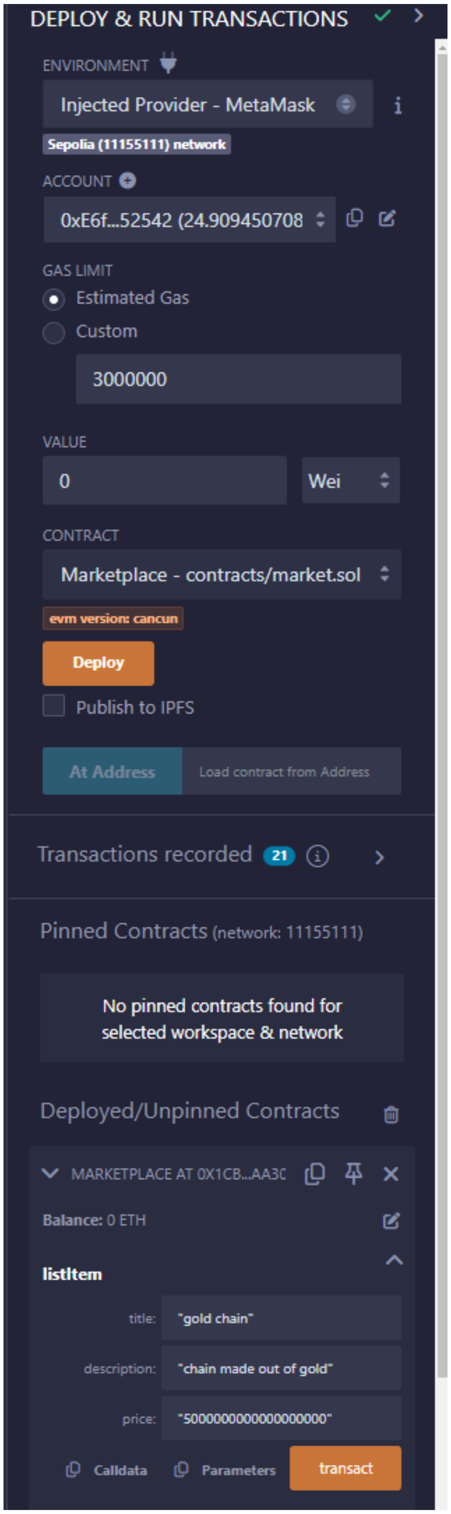


```
0: tuple(uint256,address,address,string,strin
g,uint256,bool)[]: 0,0xE6f47AA75ACDF226
D9F1C356F955ff6Bb8552542,0xc6136f8C1
337728fb7Cba5efc72F83C6128841Ac,roc
k,best rock in oklahoma,10000000000000
00000,true
```

TRUE!

Test 6: Account 1 lists an item outside of Account 2’s balance and Account 2 tries to buy the item with insufficient funds.

Account 1 lists an item for 5000000000000000000 WEI or 5 ETH.



DEPLOY & RUN TRANSACTIONS

ENVIRONMENT

Injected Provider - MetaMask

Sepolia (11155111) network

ACCOUNT

0xE6f...52542 (24.909450708

GAS LIMIT

● Estimated Gas

○ Custom

3000000

VALUE

0 Wei

CONTRACT

Marketplace - contracts/market.sol

evm version: cancun

Deploy
☐ Publish to IPFS
At Address
Load contract from Address
Transactions recorded 21
Pinned Contracts (network: 11155111)

No pinned contracts found for selected workspace & network

Deployed/Unpinned Contracts

MARKETPLACE AT 0X1CB...AA3C

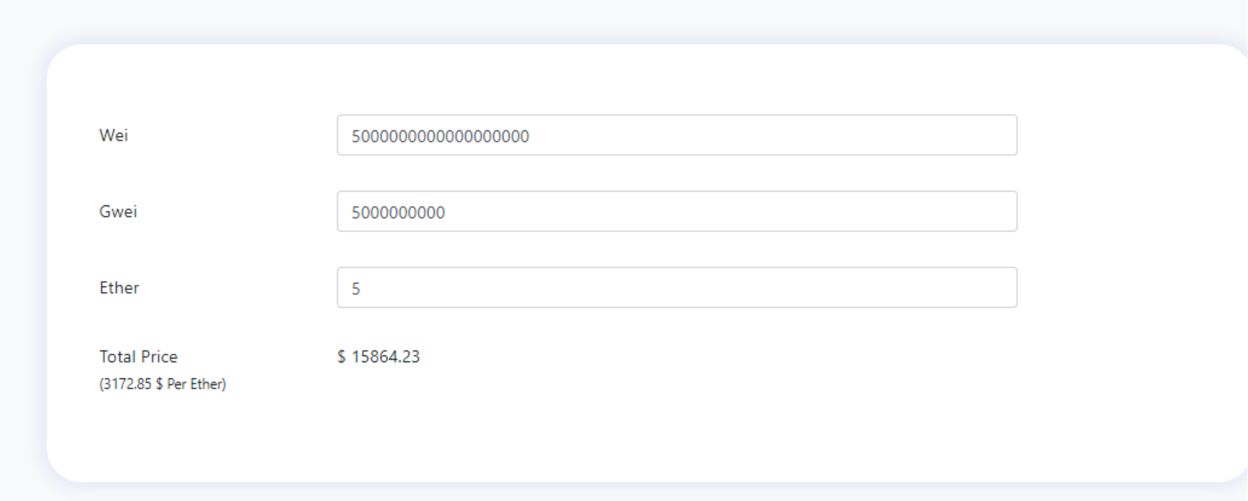
Balance: 0 ETH

listItem

title: "gold chain"

description: "chain made out of gold"

price: "5000000000000000000"

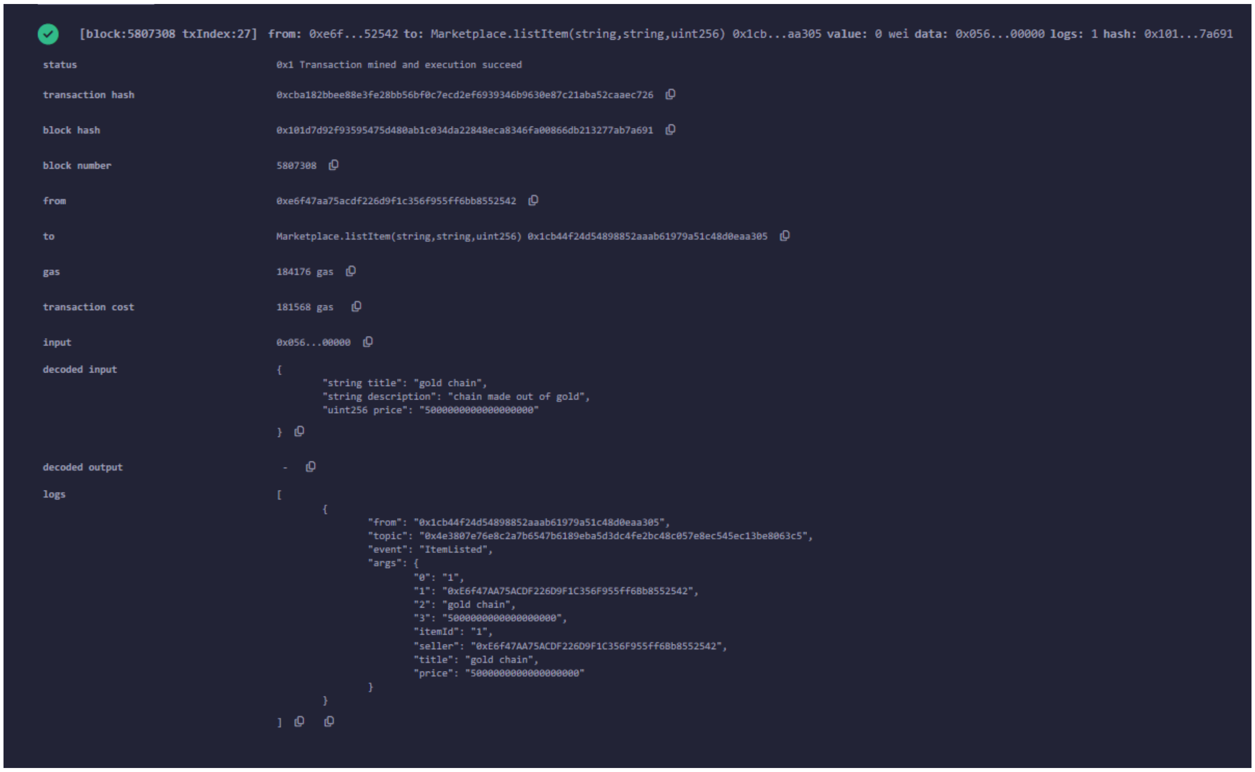


Wei 5000000000000000000

Gwei 5000000000

Ether 5

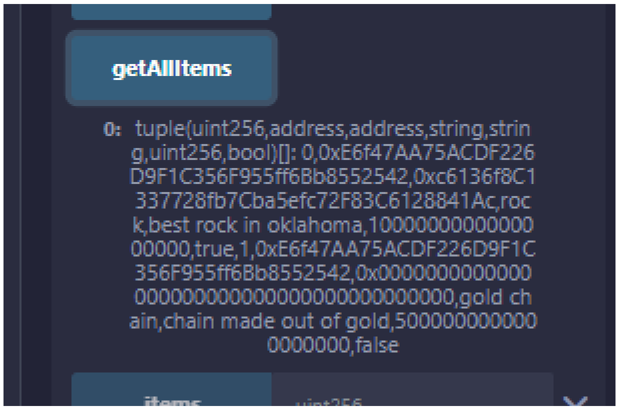
Total Price $ 15864.23
(3172.85 $ Per Ether)



```
[block:5807308 txIndex:27] from: 0xe6f...52542 to: Marketplace.listItem(string,string,uint256) 0x1cb...aa305 value: 0 wei data: 0x056...00000 logs: 1 hash: 0x101...7a691
status                  0x1 Transaction mined and execution succeed
transaction hash        0xcba182bbee88e3fe28bb56bf0c7ecd2ef6939346b9630e87c21aba52caaec726
block hash              0x101d7d92f93595475d480ab1c034da22848eca8346fa00866db213277ab7a691
block number            5807308
from                    0xe6f47aa75acdf226d9f1c356f955ff6bb8552542
to                      Marketplace.listItem(string,string,uint256) 0x1cb44f24d54898852aaab61979a51c48d0eaa305
gas                     184176 gas
transaction cost        181568 gas
input                   0x056...00000
decoded input           {
                                "string title": "gold chain",
                                "string description": "chain made out of gold",
                                "uint256 price": "5000000000000000000"
                        }
decoded output           -
logs                    [
                                {
                                        "from": "0x1cb44f24d54898852aaab61979a51c48d0eaa305",
                                        "topic": "0x4e3807e76e8c2a7b6547b6189eba5d3dc4fe2bc48c057e8ec545ec13be8063c5",
                                        "event": "ItemListed",
                                        "args": {
                                                "0": "1",
                                                "1": "0xE6f47AA75ACDF226D9F1C356F955ff6Bb8552542",
                                                "2": "gold chain",
                                                "3": "5000000000000000000",
                                                "itemId": "1",
                                                "seller": "0xE6f47AA75ACDF226D9F1C356F955ff6Bb8552542",
                                                "title": "gold chain",
                                                "price": "5000000000000000000"
                                        }
                                }
                        ]
```

Item successfully listed. Id item = 1

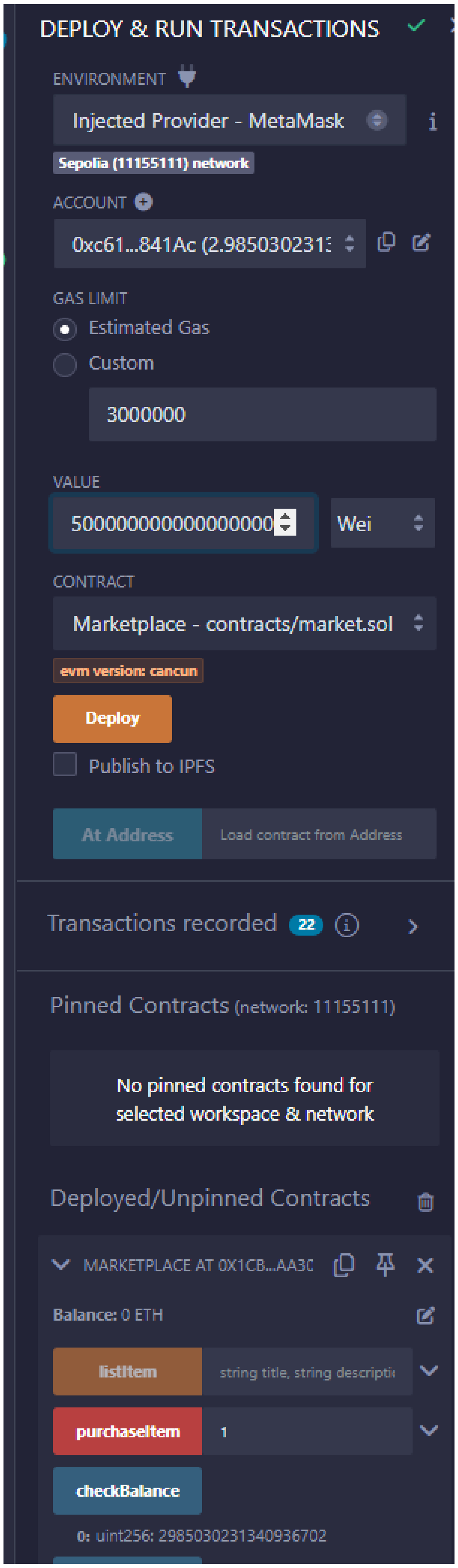
Call getAllItems to confirm listing and not sold yet:



```
0:  tuple(uint256,address,address,string,strin
    g,uint256,bool)[]: 0,0xE6f47AA75ACDF226
    D9F1C356F955ff6Bb8552542,0xc6136f8C1
    337728fb7Cba5efc72F83C6128841Ac,roc
    k,best rock in oklahoma,10000000000000
    00000,true,1,0xE6f47AA75ACDF226D9F1C
    356F955ff6Bb8552542,0x0000000000000
    000000000000000000000000000,gold ch
    ain,chain made out of gold,500000000000
    0000000,false
```

TRUE

Account 2 logs in and has a balance less than the listed price of the item they are trying to buy. Balance = 2985030231340936702 WEI and Listed item is 5000000000000000000 WEI.



DEPLOY & RUN TRANSACTIONS

ENVIRONMENT: Injected Provider - MetaMask

Sepolia (11155111) network

ACCOUNT: 0xc61...841Ac (2.985030231

GAS LIMIT

● Estimated Gas

○ Custom

3000000

VALUE: 500000000000000000 Wei

CONTRACT: Marketplace - contracts/market.sol

evm version: cancun

Transactions recorded 22

Pinned Contracts (network: 11155111)

No pinned contracts found for selected workspace & network

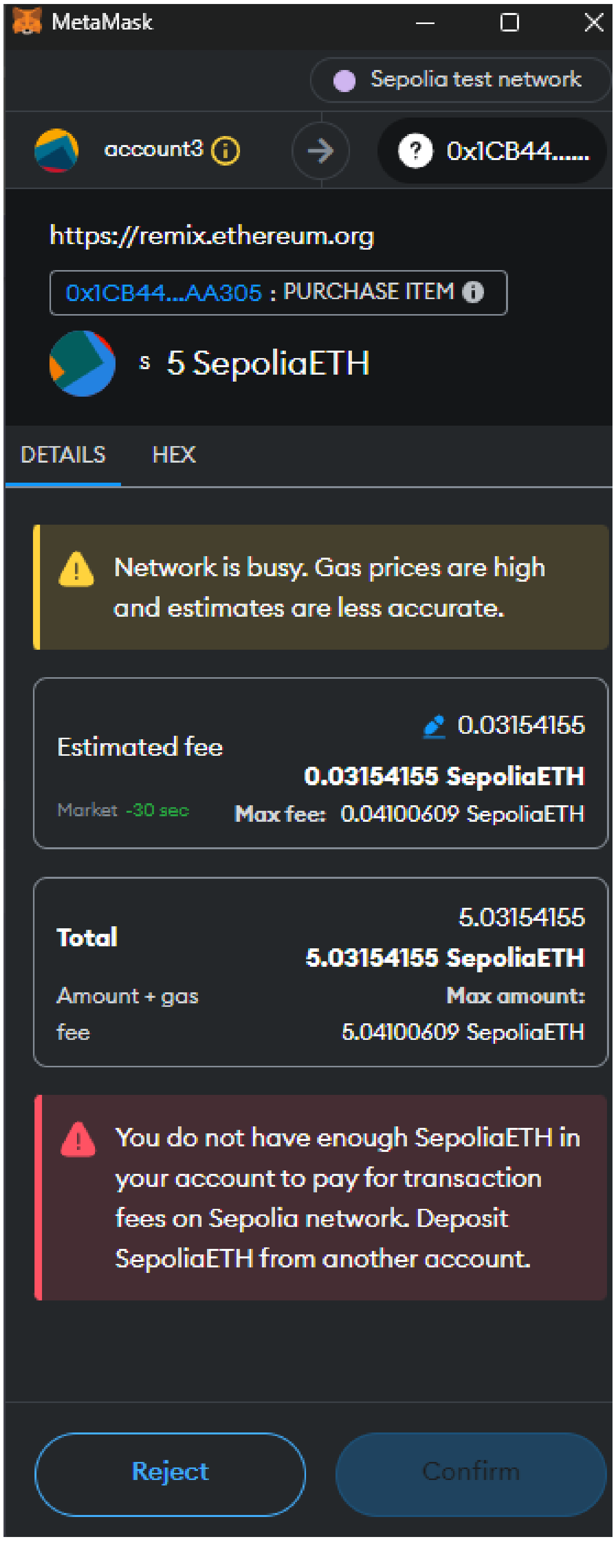
Deployed/Unpinned Contracts

MARKETPLACE AT 0X1CB...AA30

Balance: 0 ETH

purchaseItem: 1

0: uint256: 2985030231340936702



MetaMask

Sepolia test network

account3 → 0x1CB44......

https://remix.ethereum.org

0x1CB44...AA305 : PURCHASE ITEM

5 SepoliaETH

Network is busy. Gas prices are high and estimates are less accurate.

Estimated fee 0.03154155
0.03154155 SepoliaETH
Market ~30 sec Max fee: 0.04100609 SepoliaETH

Total 5.03154155
5.03154155 SepoliaETH
Amount + gas fee Max amount: 5.04100609 SepoliaETH

You do not have enough SepoliaETH in your account to pay for transaction fees on Sepolia network. Deposit SepoliaETH from another account.

The transaction can’t even be processed because Account 2 has insufficient funds.
TEST PASSED.

TEST RESULTS:

Test 1: Seller tries to sell item with 0 Wei = PASS

Test 2: Seller sells item with over 1 Wei = PASS

Test 3: Buyer tries to buy their own listed items = PASS

Test 4: Buyer tries to buy item under the list items value = PASS

Test 5: Buyer tries to buy the listed item equal to the listed item's value = PASS

Test 6: Account 1 lists an item outside of Account 2's balance and Account 2 tries to buy the item with insufficient funds = PASS

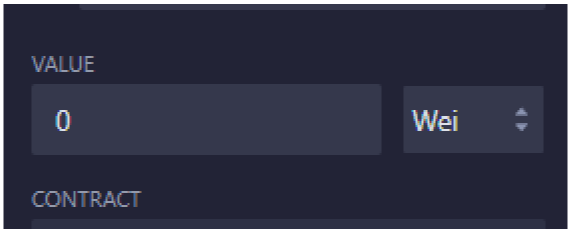
Challenges and Solutions:

1. **Lack of understanding and overcomplication at the beginning with entering listing price:** Towards the beginning, I had trouble understanding how the smart contracts operated. I had problems using Dr. Maiti's banking sample code onto my marketplace contract. I eventually realized that I was inputting the value field wrong. I overthought and thought that I needed to input a value argument in the purchase functions. This caused issues down the lines when trying to purchase listed items.

   Ex:

   listItem(string memory title, string memory description, uintt price)
   purchaseItem(uint itemId, unit price)

   Solution: By adding the price parameter into the purchaseItem function, the purchasing price was not matching up with the listing price via variable. I solved this by keeping it simple (just purchase with the itemId since I could retrieve the listing price via itemId) and figured out that when listing item prices, you enter the value of the price using

   VALUE
   0 Wei
   CONTRACT

   Used: purchaseItem(uint itemId)

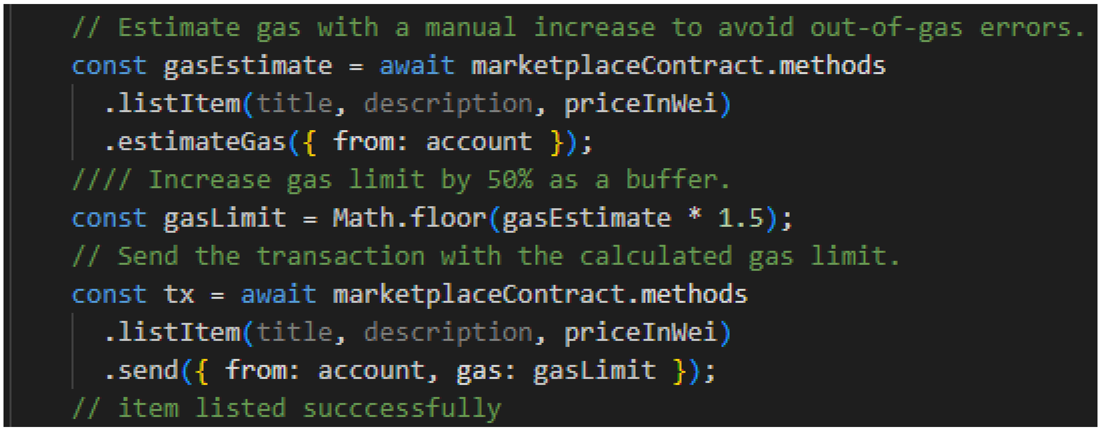
   (I was dumb lol)

2. **Conversion problems:** I had a problem I couldn't list Sepolia ETH prices down and used WEI prices instead, but on the Remix, the contract would only work if it was in WEI. This caused some compilations with the buyer getting WEI amount ETH prices. (Ex: Seller list item 2000000000000000000 WEI and the buyer getting a transaction price of 2000000000000000000 ETH). Obviously, this is wrong.

   Solution: Use ETH prices and convert to WEI on app.js so that when the buyer tries to buy the item, they got the correct amount to pay.

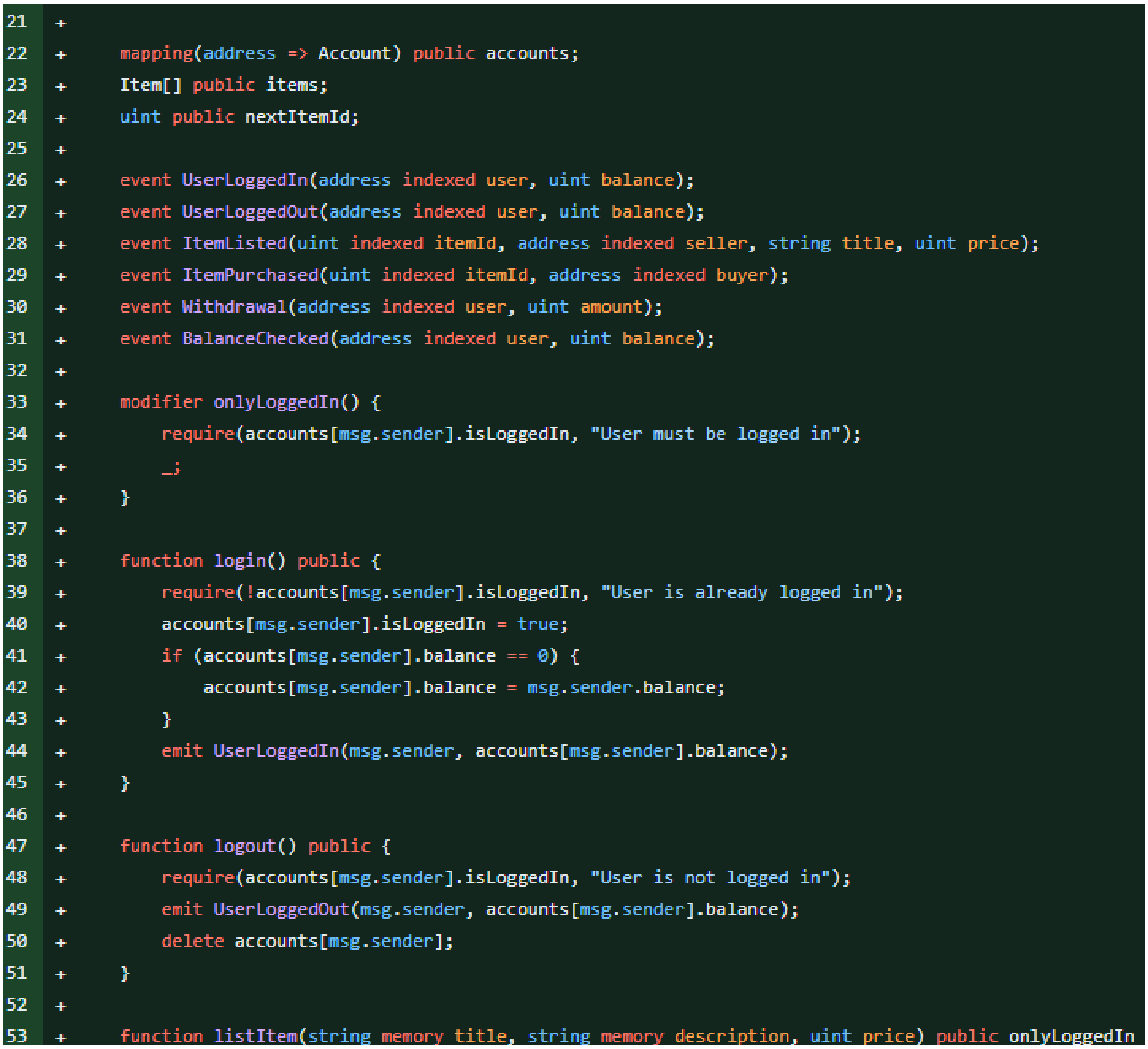
3. **Transaction gas fee:** I had a problem where the transaction was not going through when listing items because I had a little to no transaction fee.

   Solution: On the listingItem function, set gasLimit fee by multiplying the gasEstimate multiplied by 50%. This solved the transaction gas fee issue and made the listing transaction go through.



```
// Estimate gas with a manual increase to avoid out-of-gas errors.
const gasEstimate = await marketplaceContract.methods
  .listItem(title, description, priceInWei)
  .estimateGas({ from: account });
//// Increase gas limit by 50% as a buffer.
const gasLimit = Math.floor(gasEstimate * 1.5);
// Send the transaction with the calculated gas limit.
const tx = await marketplaceContract.methods
  .listItem(title, description, priceInWei)
  .send({ from: account, gas: gasLimit });
// item listed succcessfully
```

4. **No need for login and logout:** I initially thought that you needed to change wallet address on MetaMask by implementing a login and logout function (shown in history of Adding contract version 2 commit). This caused complications with the wallet addresses switching and listing and purchase transactions. This was just due to overcomplication on my part.



```
+
+    mapping(address => Account) public accounts;
+    Item[] public items;
+    uint public nextItemId;
+
+    event UserLoggedIn(address indexed user, uint balance);
+    event UserLoggedOut(address indexed user, uint balance);
+    event ItemListed(uint indexed itemId, address indexed seller, string title, uint price);
+    event ItemPurchased(uint indexed itemId, address indexed buyer);
+    event Withdrawal(address indexed user, uint amount);
+    event BalanceChecked(address indexed user, uint balance);
+
+    modifier onlyLoggedIn() {
+        require(accounts[msg.sender].isLoggedIn, "User must be logged in");
+        _;
+    }
+
+    function login() public {
+        require(!accounts[msg.sender].isLoggedIn, "User is already logged in");
+        accounts[msg.sender].isLoggedIn = true;
+        if (accounts[msg.sender].balance == 0) {
+            accounts[msg.sender].balance = msg.sender.balance;
+        }
+        emit UserLoggedIn(msg.sender, accounts[msg.sender].balance);
+    }
+
+    function logout() public {
+        require(accounts[msg.sender].isLoggedIn, "User is not logged in");
+        emit UserLoggedOut(msg.sender, accounts[msg.sender].balance);
+        delete accounts[msg.sender];
+    }
+
+    function listItem(string memory title, string memory description, uint price) public onlyLoggedIn
```

Solution: User can change wallets via MetaMask browser extension