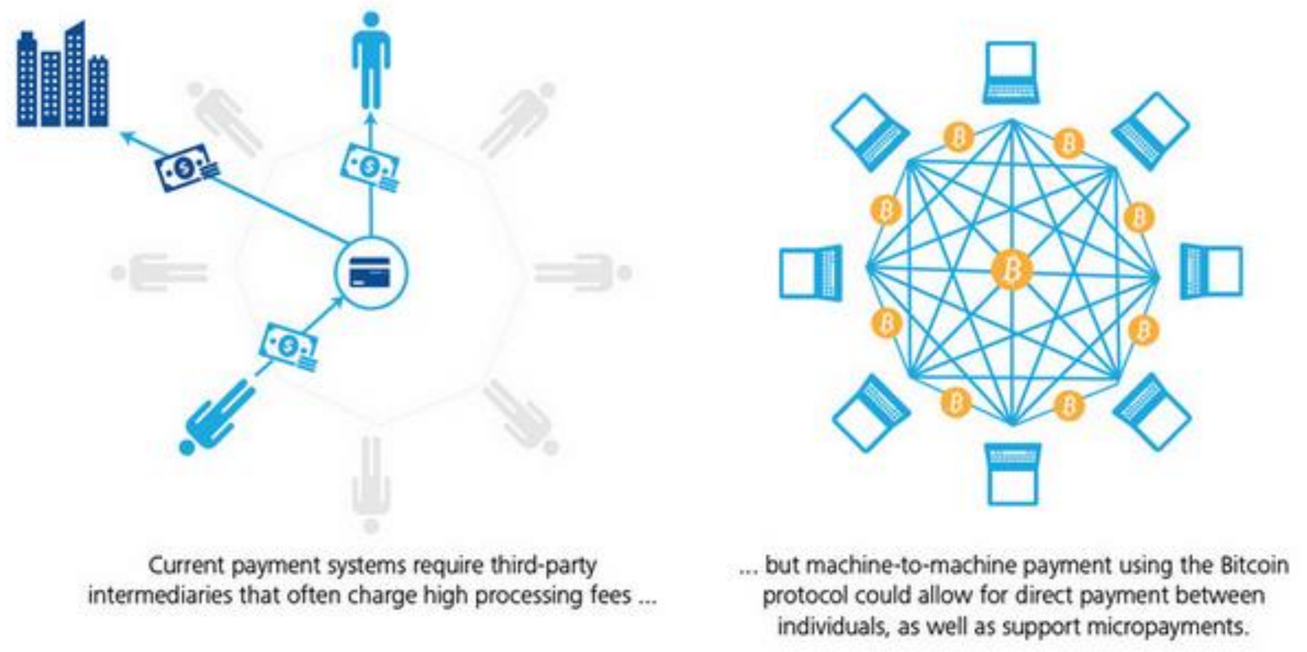
DAPP with privacy mechanisms in Ethereum

Eloy Ramirez Hernanz

Agenda

- Privacy in Blockchain
- Shopping List implementation (Use case)

Privacy in Blockchain


Current payment systems require third-party intermediaries that often charge high processing fees ...

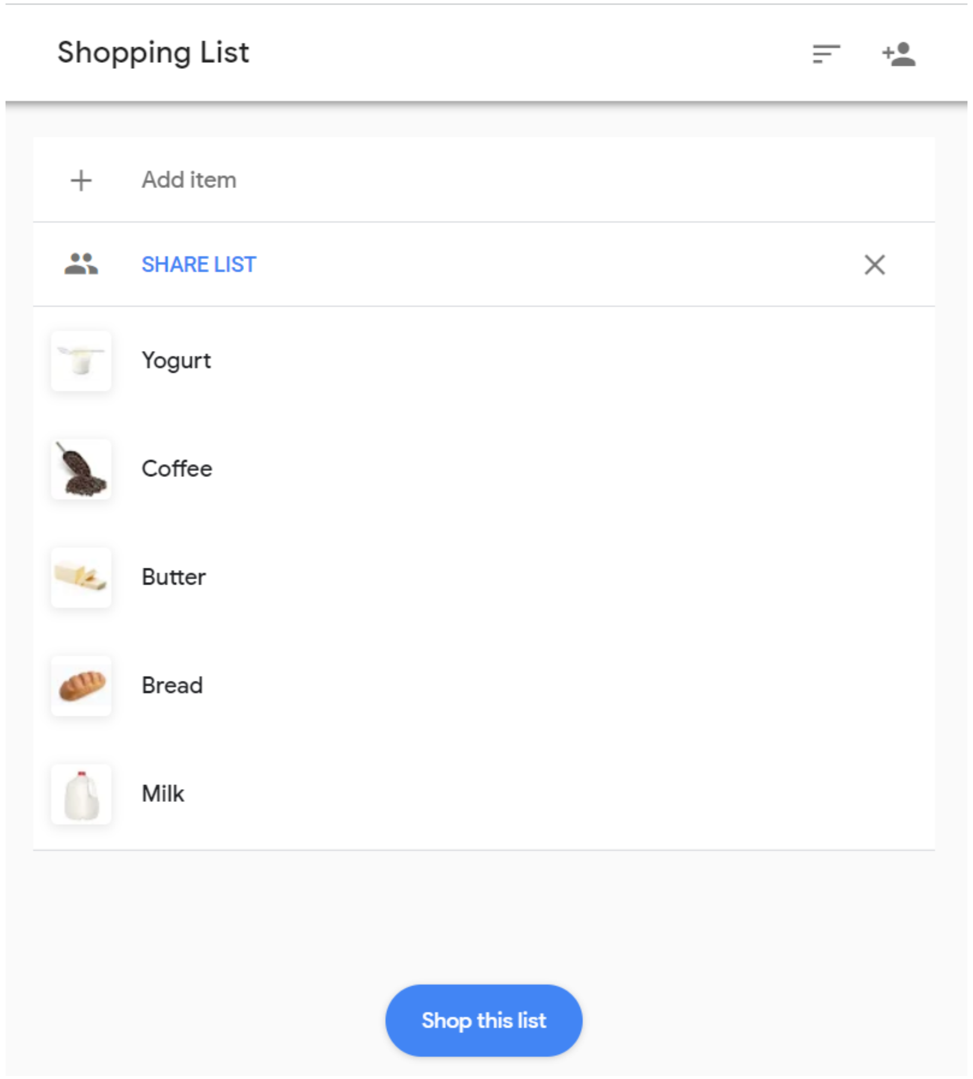
... but machine-to-machine payment using the Bitcoin protocol could allow for direct payment between individuals, as well as support micropayments.

- Advance in different fields
- Avoid third party
- Distributed ledger

Privacy in Blockchain

- The new data is added to the blockchain in form of transactions, which after being approved by the nodes of the blockchain, is added to the network in form of blocks.
- As all the nodes have a copy of the blockchain, they can access to the information added.
- It is important not adding relevant information in the Smart Contracts in spite of the type used for the variable which has the information (public, private or secret).

Shopping List Implementation

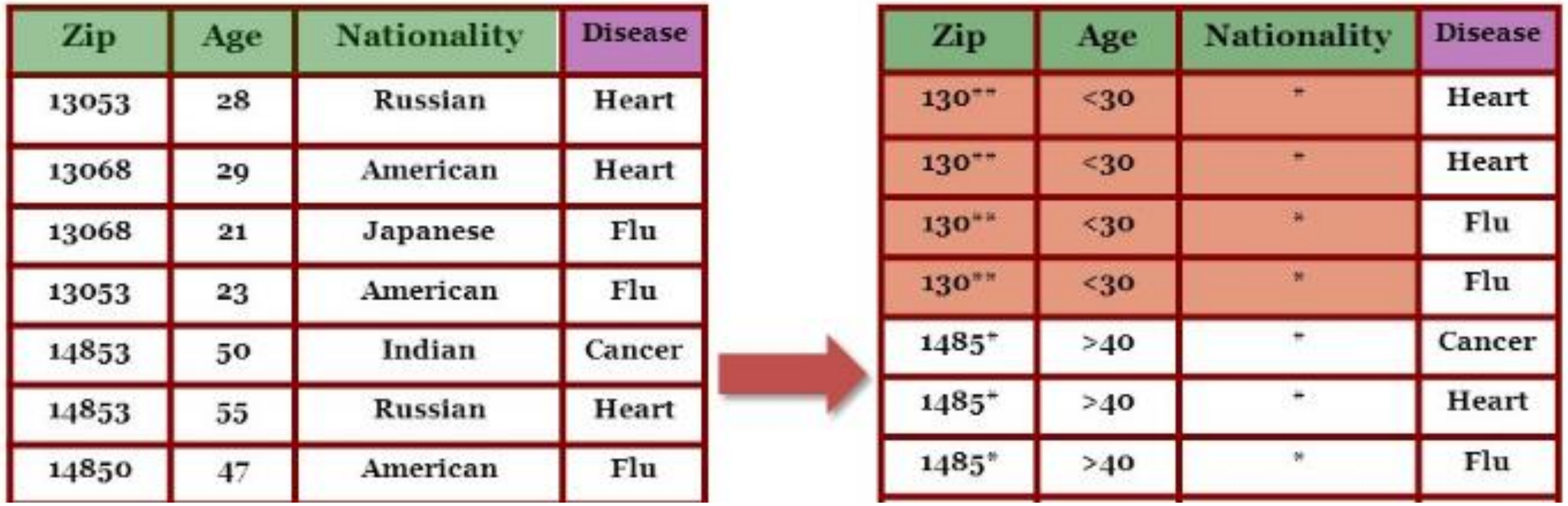


- Google has a web application for the shopping list (same as other supermarkets)
- They offer you the option of shopping this list
- Is this information used by Google? Maybe some user would prefer to omit some items, or even all of them

Shopping List Implementation

- DAPP, decentralized application because it is used with a blockchain, Ethereum in this case
- Therefore each node participating in this Application has access to the blockchain (the state), and is able to check the information. Supermarkets want this information for selling to brands or obtaining value from them
- Avoid third party managing info and maintain privacy by using K-anonymity, avoiding being identified by others

Shopping List Implementation

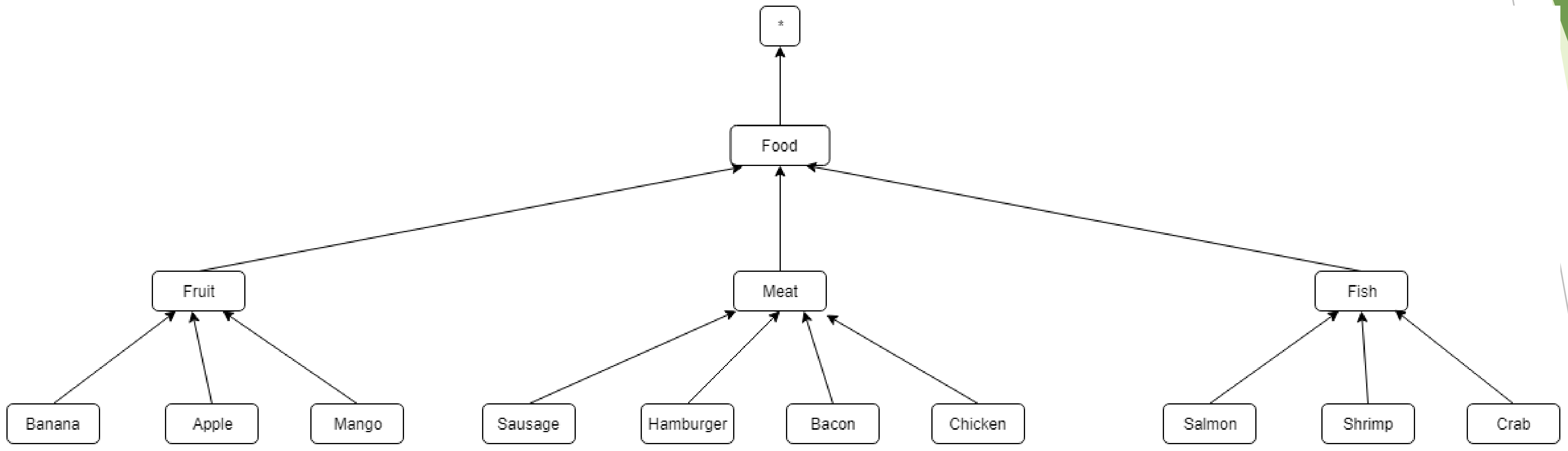


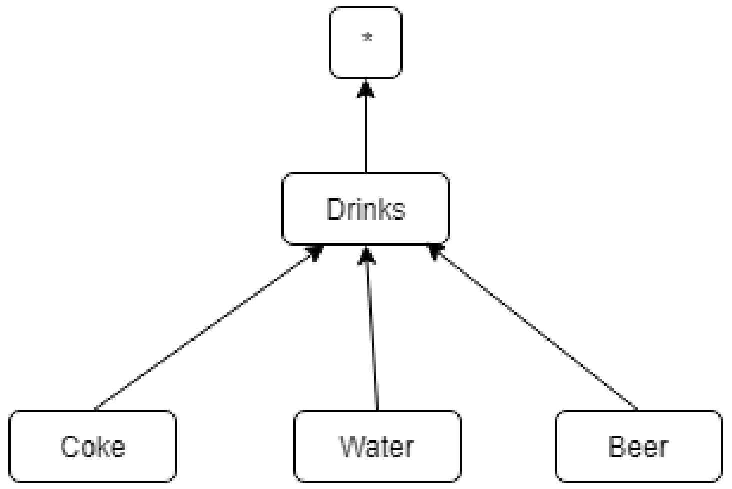
Zip	Age	Nationality	Disease
13053	28	Russian	Heart
13068	29	American	Heart
13068	21	Japanese	Flu
13053	23	American	Flu
14853	50	Indian	Cancer
14853	55	Russian	Heart
14850	47	American	Flu

Zip	Age	Nationality	Disease
130**	<30	*	Heart
130**	<30	*	Heart
130**	<30	*	Flu
130**	<30	*	Flu
1485*	>40	*	Cancer
1485*	>40	*	Heart
1485*	>40	*	Flu

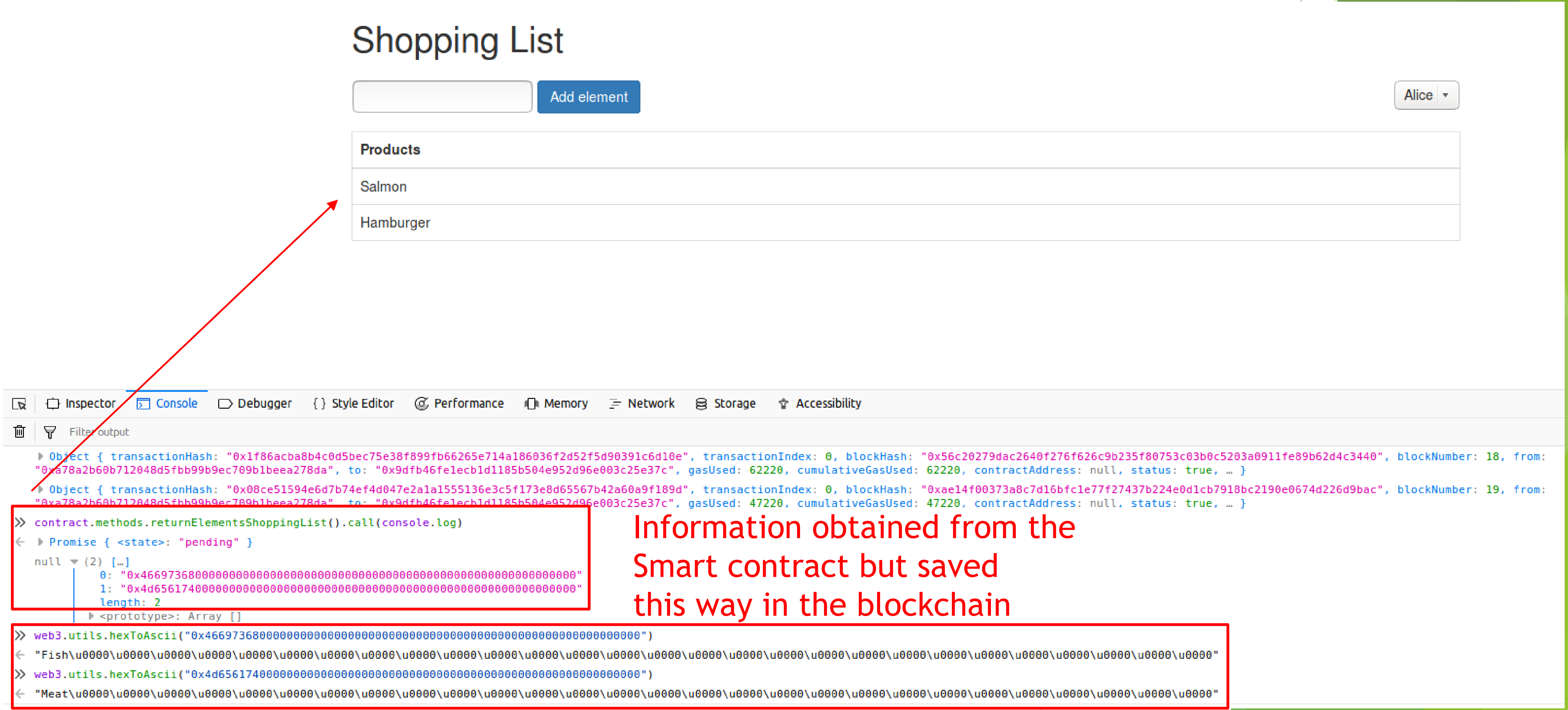
Example of k-anonymity

Shopping List Implementation





Shopping List Implementation

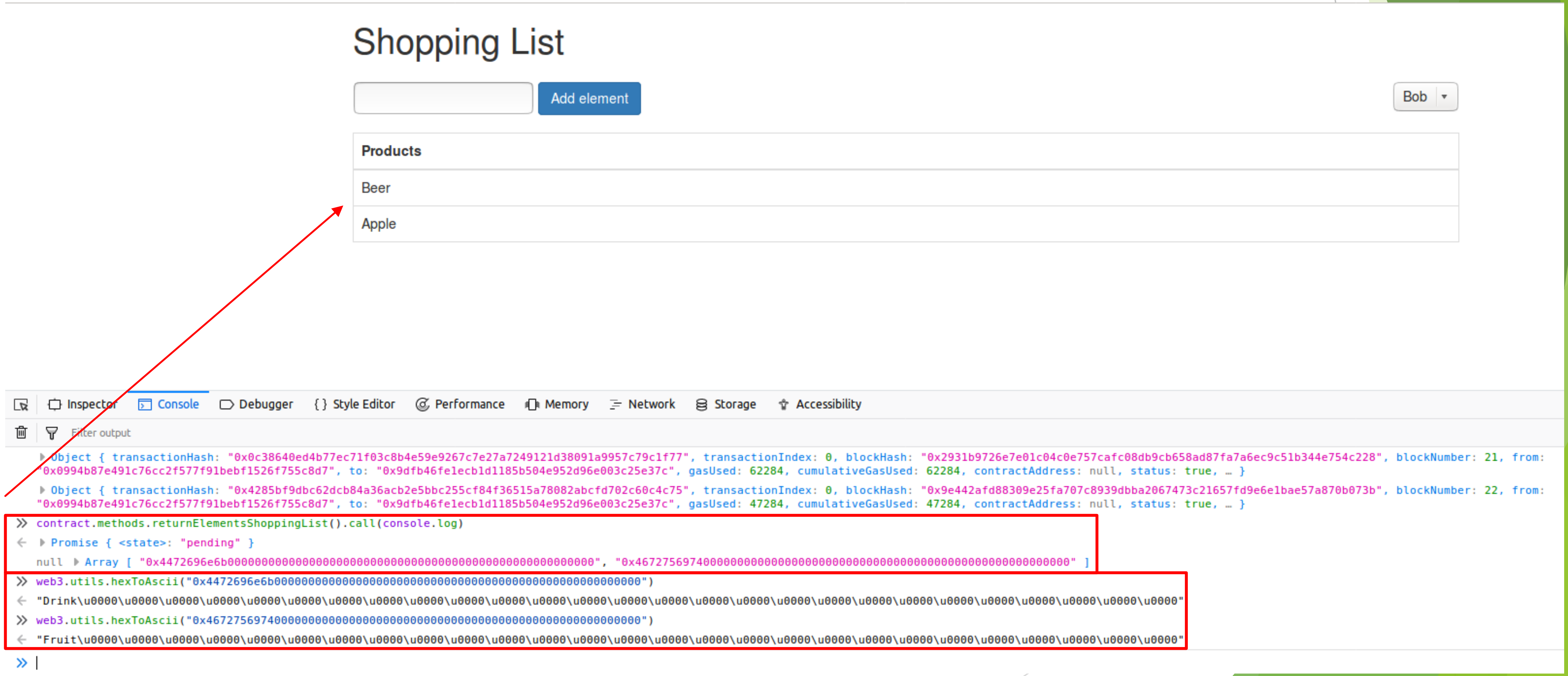


Products
Salmon
Hamburger

```
▶ Object { transactionHash: "0x1f86acba8b4c0d5bec75e38f899fb66265e714a186036f2d52f5d90391c6d10e", transactionIndex: 0, blockHash: "0x56c20279dac2640f276f626c9b235f80753c03b0c5203a0911fe89b62d4c3440", blockNumber: 18, from:
"0xa78a2b60b712048d5fbb99b9ec709b1beea278da", to: "0x9dfb46fe1ecb1d1185b504e952d96e003c25e37c", gasUsed: 62220, cumulativeGasUsed: 62220, contractAddress: null, status: true, … }
▶ Object { transactionHash: "0x08ce51594e6d7b74ef4d047e2a1a1555136e3c5f173e8d65567b42a60a9f189d", transactionIndex: 0, blockHash: "0xae14f00373a8c7d16bfc1e77f27437b224e0d1cb7918bc2190e0674d226d9bac", blockNumber: 19, from:
"0xa78a2b60b712048d5fbb99b9ec709b1beea278da", to: "0x9dfb46fe1ecb1d1185b504e952d96e003c25e37c", gasUsed: 47220, cumulativeGasUsed: 47220, contractAddress: null, status: true, … }
>> contract.methods.returnElementsShoppingList().call(console.log)
← ▶ Promise { <state>: "pending" }
  null ▼ (2) […]
           0: "0x4669736800000000000000000000000000000000000000000000000000000000"
           1: "0x4d65617400000000000000000000000000000000000000000000000000000000"
           length: 2
         ▶ <prototype>: Array []
>> web3.utils.hexToAscii("0x4669736800000000000000000000000000000000000000000000000000000000")
← "Fish\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000"
>> web3.utils.hexToAscii("0x4d65617400000000000000000000000000000000000000000000000000000000")
← "Meat\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000"
```

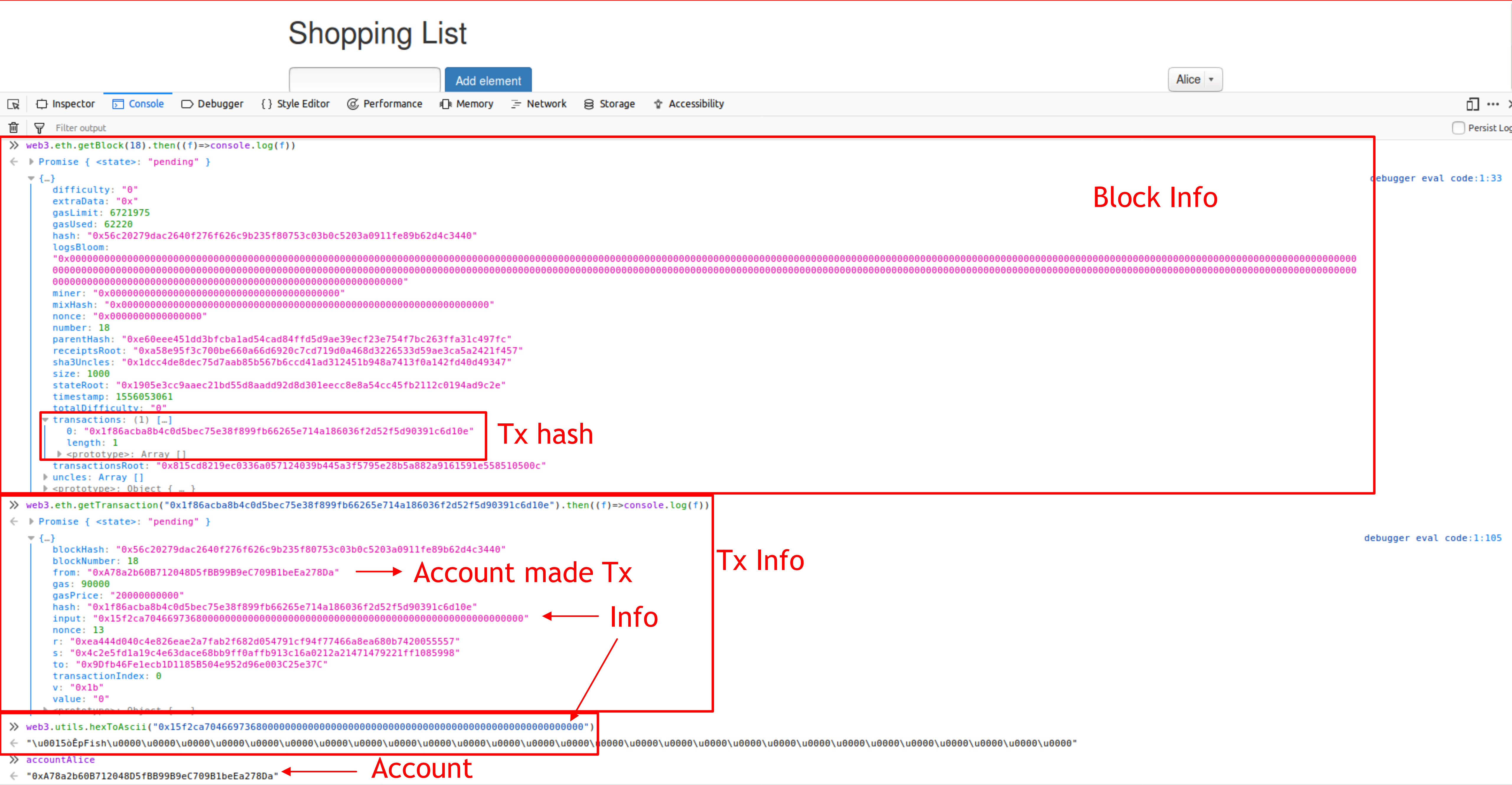
Information obtained from the Smart contract but saved this way in the blockchain

Shopping List Implementation



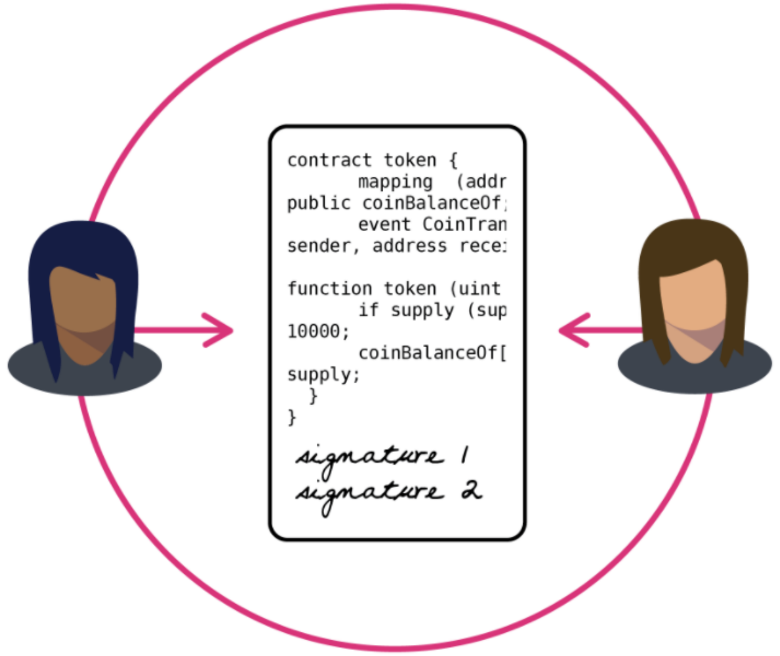
Products
Beer
Apple

```
▸ Object { transactionHash: "0x0c38640ed4b77ec71f03c8b4e59e9267c7e27a7249121d38091a9957c79c1f77", transactionIndex: 0, blockHash: "0x2931b9726e7e01c04c0e757cafc08db9cb658ad87fa7a6ec9c51b344e754c228", blockNumber: 21, from:
"0x0994b87e491c76cc2f577f91bebf1526f755c8d7", to: "0x9dfb46fe1ecb1d1185b504e952d96e003c25e37c", gasUsed: 62284, cumulativeGasUsed: 62284, contractAddress: null, status: true, … }
▸ Object { transactionHash: "0x4285bf9dbc62dcb84a36acb2e5bbc255cf84f36515a78082abcfd702c60c4c75", transactionIndex: 0, blockHash: "0x9e442afd88309e25fa707c8939dbba2067473c21657fd9e6e1bae57a870b073b", blockNumber: 22, from:
"0x0994b87e491c76cc2f577f91bebf1526f755c8d7", to: "0x9dfb46fe1ecb1d1185b504e952d96e003c25e37c", gasUsed: 47284, cumulativeGasUsed: 47284, contractAddress: null, status: true, … }
>> contract.methods.returnElementsShoppingList().call(console.log)
← ▸ Promise { <state>: "pending" }
null ▸ Array [ "0x4472696e6b000000000000000000000000000000000000000000000000000000", "0x4672756974000000000000000000000000000000000000000000000000000000" ]
>> web3.utils.hexToAscii("0x4472696e6b000000000000000000000000000000000000000000000000000000")
← "Drink\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000"
>> web3.utils.hexToAscii("0x4672756974000000000000000000000000000000000000000000000000000000")
← "Fruit\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000"
>> |
```



```
>> web3.eth.getBlock(18).then((f)=>console.log(f))
<- ▶ Promise { <state>: "pending" }
   ▼ {…}                                                                                    debugger eval code:1:33
      difficulty: "0"
      extraData: "0x"
      gasLimit: 6721975
      gasUsed: 62220
      hash: "0x56c20279dac2640f276f626c9b235f80753c03b0c5203a0911fe89b62d4c3440"
      logsBloom:
      "0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
      00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
      00000000000000000000000000000000000000000000000000000000000000"
      miner: "0x0000000000000000000000000000000000000000"
      mixHash: "0x0000000000000000000000000000000000000000000000000000000000000000"
      nonce: "0x0000000000000000"
      number: 18
      parentHash: "0xe60eee451dd3bfcba1ad54cad84ffd5d9ae39ecf23e754f7bc263ffa31c497fc"
      receiptsRoot: "0xa58e95f3c700be660a66d6920c7cd719d0a468d3226533d59ae3ca5a2421f457"
      sha3Uncles: "0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347"
      size: 1000
      stateRoot: "0x1905e3cc9aaec21bd55d8aadd92d8d301eecc8e8a54cc45fb2112c0194ad9c2e"
      timestamp: 1556053061
      totalDifficulty: "0"
    ▼ transactions: (1) […]
        0: "0x1f86acba8b4c0d5bec75e38f899fb66265e714a186036f2d52f5d90391c6d10e"
        length: 1
      ▶ <prototype>: Array []
      transactionsRoot: "0x815cd8219ec0336a057124039b445a3f5795e28b5a882a9161591e558510500c"
    ▶ uncles: Array []
    ▶ <prototype>: Object { … }
>> web3.eth.getTransaction("0x1f86acba8b4c0d5bec75e38f899fb66265e714a186036f2d52f5d90391c6d10e").then((f)=>console.log(f))
<- ▶ Promise { <state>: "pending" }
   ▼ {…}                                                                                    debugger eval code:1:105
      blockHash: "0x56c20279dac2640f276f626c9b235f80753c03b0c5203a0911fe89b62d4c3440"
      blockNumber: 18
      from: "0xA78a2b60B712048D5fBB99B9eC709B1beEa278Da"
      gas: 90000
      gasPrice: "20000000000"
      hash: "0x1f86acba8b4c0d5bec75e38f899fb66265e714a186036f2d52f5d90391c6d10e"
      input: "0x15f2ca704669736800000000000000000000000000000000000000000000000000000000"
      nonce: 13
      r: "0xea444d040c4e826eae2a7fab2f682d054791cf94f77466a8ea680b7420055557"
      s: "0x4c2e5fd1a19c4e63dace68bb9ff0affb913c16a0212a21471479221ff1085998"
      to: "0x9Dfb46Fe1ecb1D1185B504e952d96e003C25e37C"
      transactionIndex: 0
      v: "0x1b"
      value: "0"
    ▶ <prototype>: Object { … }
>> web3.utils.hexToAscii("0x15f2ca704669736800000000000000000000000000000000000000000000000000000000")
<- "\u0015òÊpFish\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000"
>> accountAlice
<- "0xA78a2b60B712048D5fBB99B9eC709B1beEa278Da"
```

Shopping List Implementation



```
contract token {
        mapping  (addr
public coinBalanceOf;
        event CoinTran
sender, address rece:

function token (uint
        if supply (sup
10000;
        coinBalanceOf[
supply;
   }
}
```

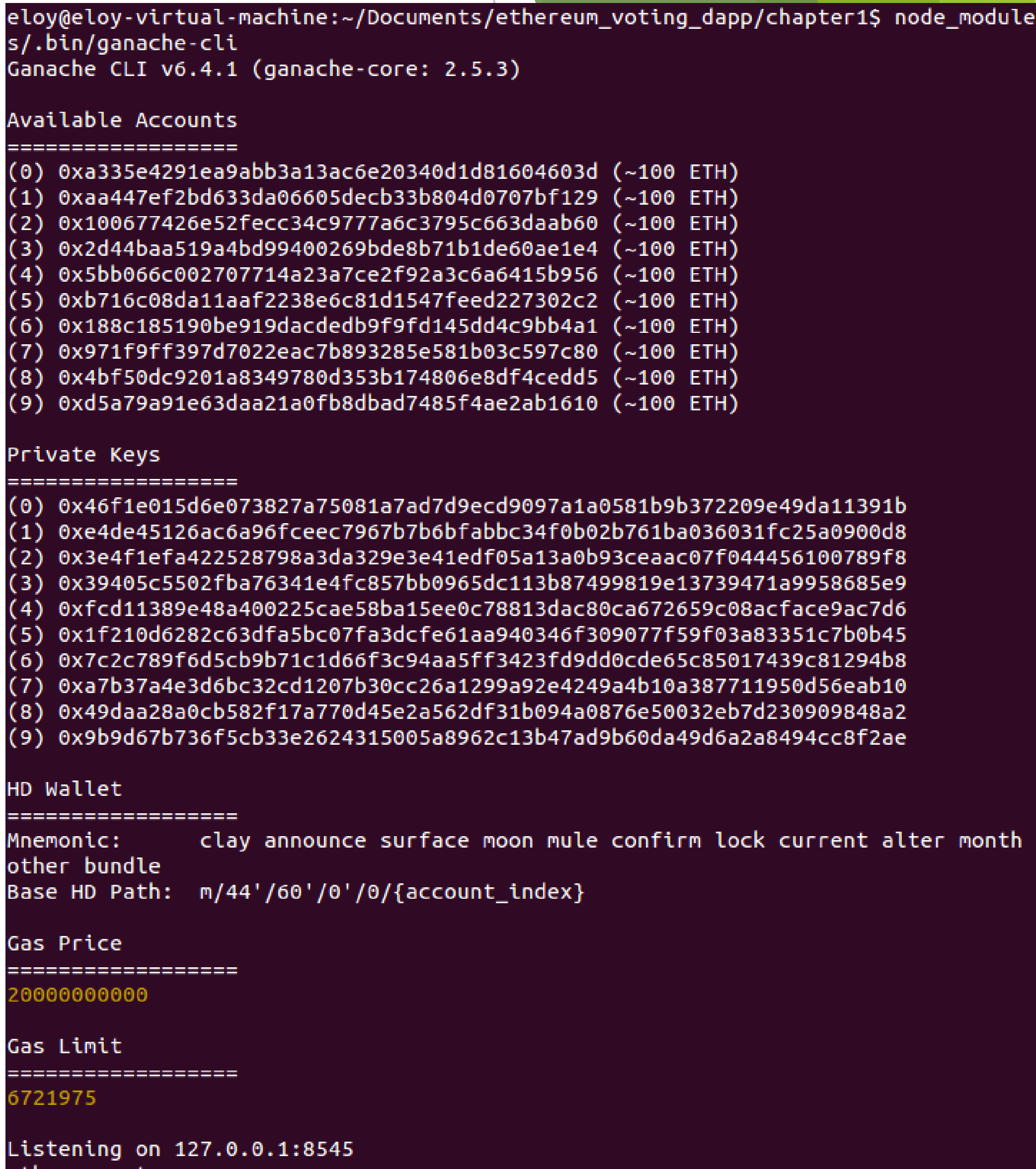
signature 1
signature 2

DAPP in Solidity

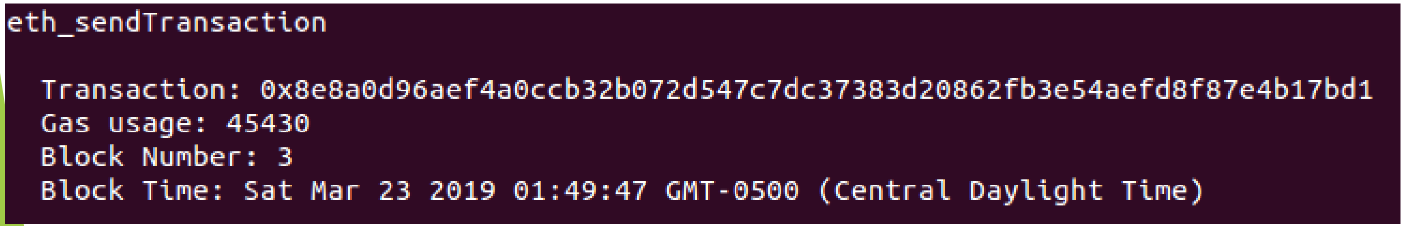
Ganache remix

Development Tools and Clients

Analysis of the Security of misprograming DAPPs: Approach to the problem

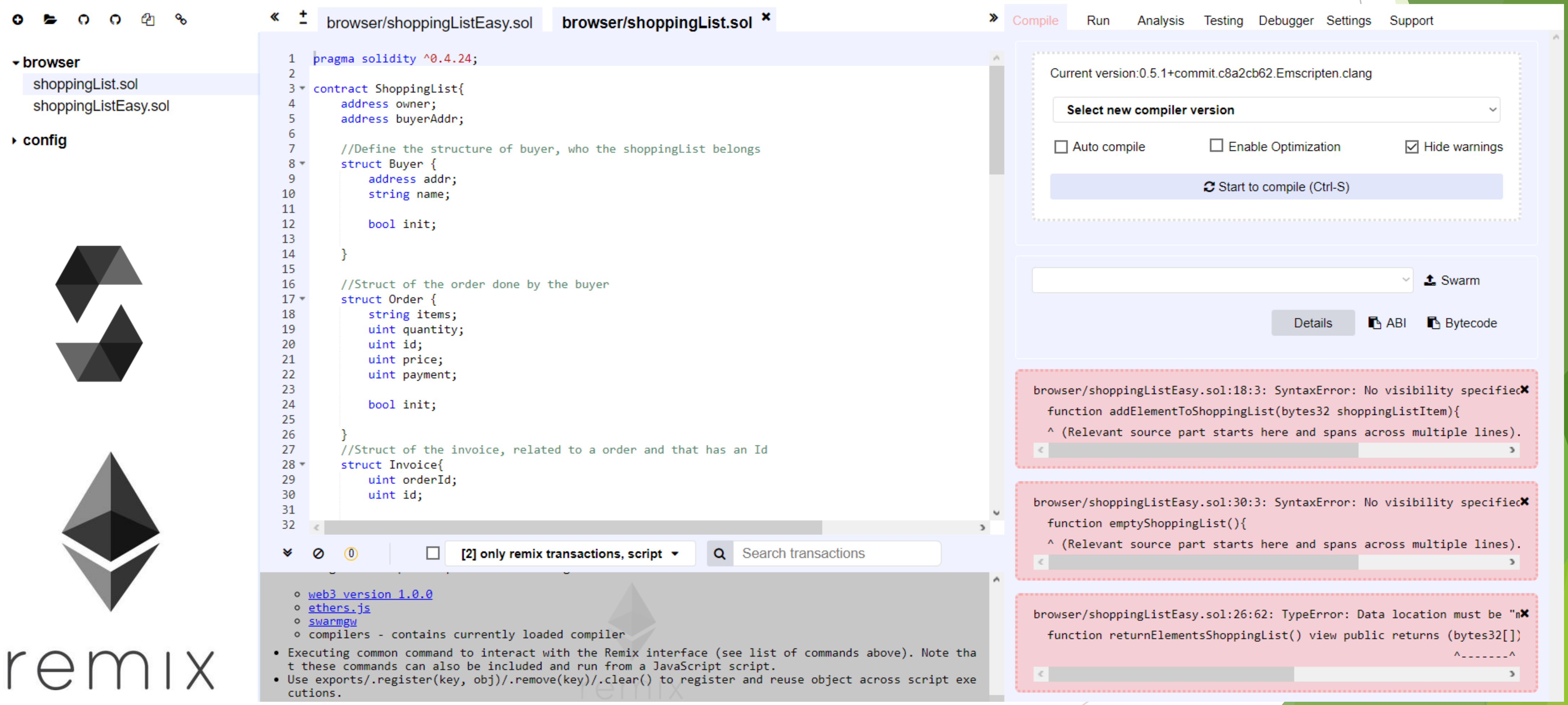


```
eloy@eloy-virtual-machine:~/Documents/ethereum_voting_dapp/chapter1$ node_module
s/.bin/ganache-cli
Ganache CLI v6.4.1 (ganache-core: 2.5.3)

Available Accounts
==================
(0) 0xa335e4291ea9abb3a13ac6e20340d1d81604603d (~100 ETH)
(1) 0xaa447ef2bd633da06605decb33b804d0707bf129 (~100 ETH)
(2) 0x100677426e52fecc34c9777a6c3795c663daab60 (~100 ETH)
(3) 0x2d44baa519a4bd99400269bde8b71b1de60ae1e4 (~100 ETH)
(4) 0x5bb066c002707714a23a7ce2f92a3c6a6415b956 (~100 ETH)
(5) 0xb716c08da11aaf2238e6c81d1547feed227302c2 (~100 ETH)
(6) 0x188c185190be919dacdedb9f9fd145dd4c9bb4a1 (~100 ETH)
(7) 0x971f9ff397d7022eac7b893285e581b03c597c80 (~100 ETH)
(8) 0x4bf50dc9201a8349780d353b174806e8df4cedd5 (~100 ETH)
(9) 0xd5a79a91e63daa21a0fb8dbad7485f4ae2ab1610 (~100 ETH)

Private Keys
==================
(0) 0x46f1e015d6e073827a75081a7ad7d9ecd9097a1a0581b9b372209e49da11391b
(1) 0xe4de45126ac6a96fceec7967b7b6bfabbc34f0b02b761ba036031fc25a0900d8
(2) 0x3e4f1efa422528798a3da329e3e41edf05a13a0b93ceaac07f044456100789f8
(3) 0x39405c5502fba76341e4fc857bb0965dc113b87499819e13739471a9958685e9
(4) 0xfcd11389e48a400225cae58ba15ee0c78813dac80ca672659c08acface9ac7d6
(5) 0x1f210d6282c63dfa5bc07fa3dcfe61aa940346f309077f59f03a83351c7b0b45
(6) 0x7c2c789f6d5cb9b71c1d66f3c94aa5ff3423fd9dd0cde65c85017439c81294b8
(7) 0xa7b37a4e3d6bc32cd1207b30cc26a1299a92e4249a4b10a387711950d56eab10
(8) 0x49daa28a0cb582f17a770d45e2a562df31b094a0876e50032eb7d230909848a2
(9) 0x9b9d67b736f5cb33e2624315005a8962c13b47ad9b60da49d6a2a8494cc8f2ae

HD Wallet
==================
Mnemonic:      clay announce surface moon mule confirm lock current alter month
other bundle
Base HD Path:  m/44'/60'/0'/0/{account_index}

Gas Price
==================
20000000000

Gas Limit
==================
6721975

Listening on 127.0.0.1:8545
```



```
eth_sendTransaction

  Transaction: 0x8e8a0d96aef4a0ccb32b072d547c7dc37383d20862fb3e54aefd8f87e4b17bd1
  Gas usage: 45430
  Block Number: 3
  Block Time: Sat Mar 23 2019 01:49:47 GMT-0500 (Central Daylight Time)
```

Shopping List Implementation



```
pragma solidity ^0.4.24;

contract ShoppingList{
    address owner;
    address buyerAddr;

    //Define the structure of buyer, who the shoppingList belongs
    struct Buyer {
        address addr;
        string name;

        bool init;

    }

    //Struct of the order done by the buyer
    struct Order {
        string items;
        uint quantity;
        uint id;
        uint price;
        uint payment;

        bool init;

    }
    //Struct of the invoice, related to a order and that has an Id
    struct Invoice{
        uint orderId;
        uint id;

```

Current version:0.5.1+commit.c8a2cb62.Emscripten.clang

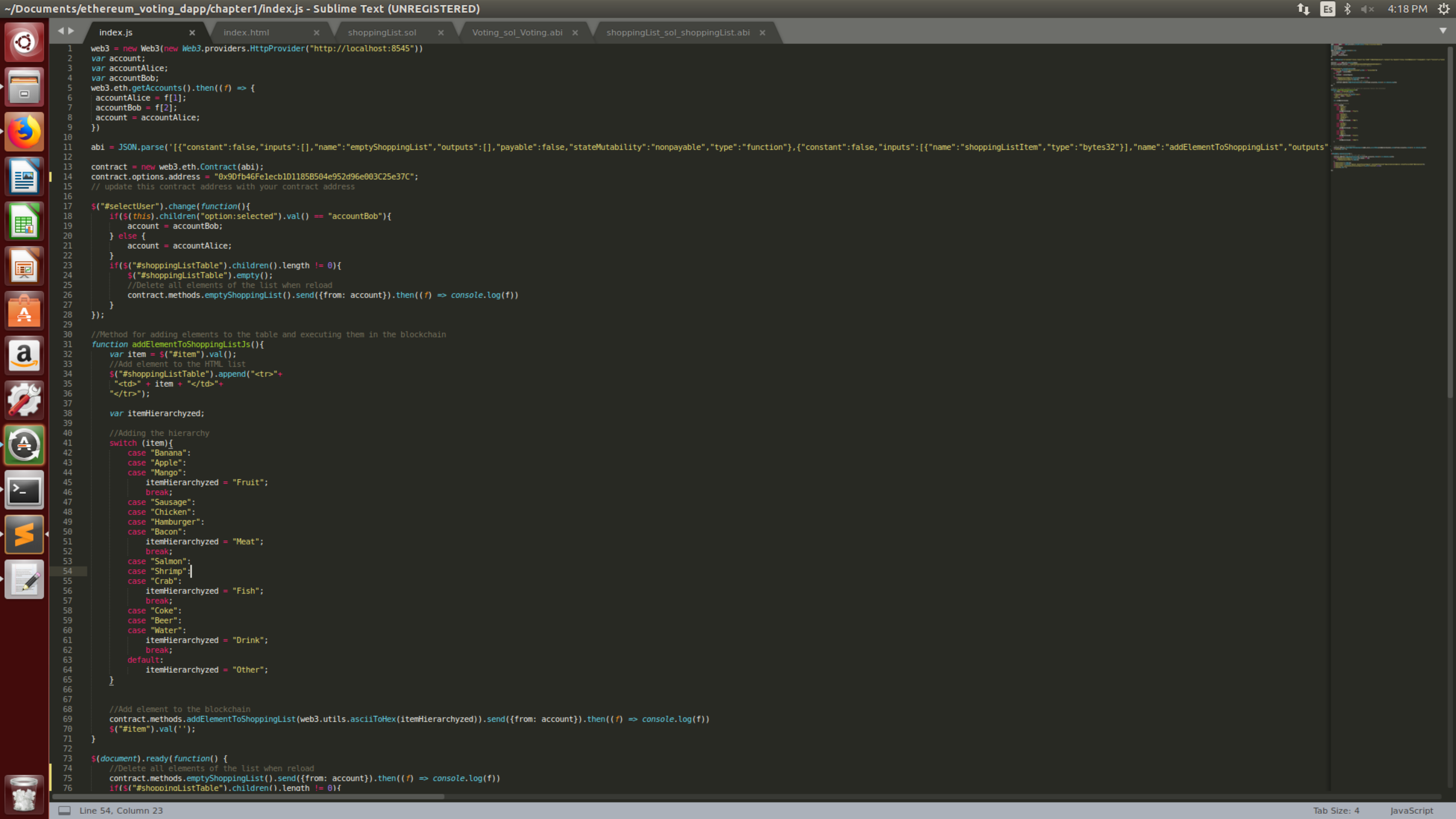
☐ Auto compile ☐ Enable Optimization ☑ Hide warnings

```
browser/shoppingListEasy.sol:18:3: SyntaxError: No visibility specifiec
  function addElementToShoppingList(bytes32 shoppingListItem){
  ^ (Relevant source part starts here and spans across multiple lines).
```

```
browser/shoppingListEasy.sol:30:3: SyntaxError: No visibility specifiec
  function emptyShoppingList(){
  ^ (Relevant source part starts here and spans across multiple lines).
```

```
browser/shoppingListEasy.sol:26:62: TypeError: Data location must be "m
  function returnElementsShoppingList() view public returns (bytes32[])
                                                            ^-------^
```

```
    o web3 version 1.0.0
    o ethers.js
    o swarmgw
    o compilers - contains currently loaded compiler
 • Executing common command to interact with the Remix interface (see list of commands above). Note tha
   t these commands can also be included and run from a JavaScript script.
 • Use exports/.register(key, obj)/.remove(key)/.clear() to register and reuse object across script exe
   cutions.
```

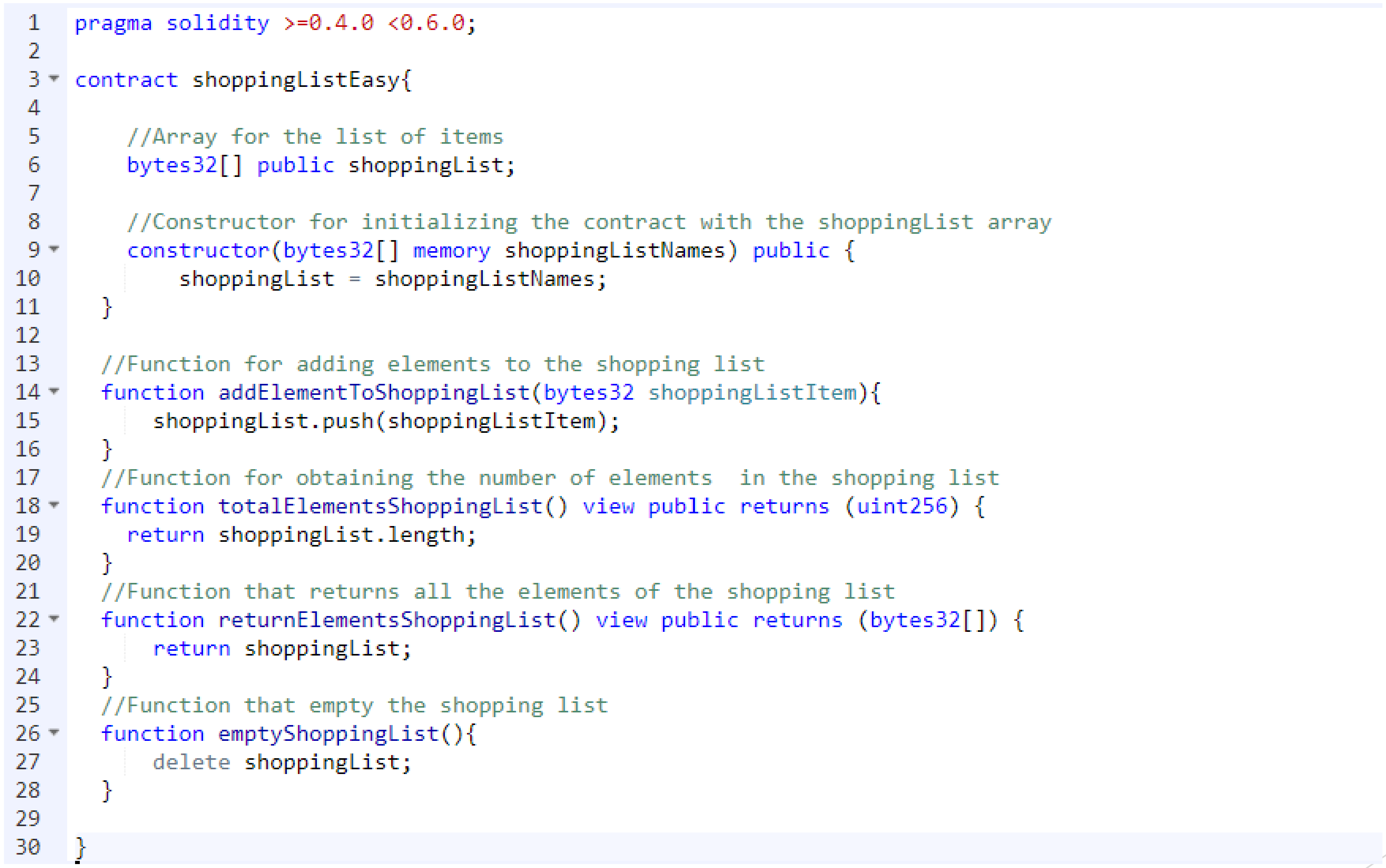


~/Documents/ethereum_voting_dapp/chapter1/index.js - Sublime Text (UNREGISTERED)

index.js | index.html | shoppingList.sol | Voting_sol_Voting.abi | shoppingList_sol_shoppingList.abi

```javascript
web3 = new Web3(new Web3.providers.HttpProvider("http://localhost:8545"))
var account;
var accountAlice;
var accountBob;
web3.eth.getAccounts().then((f) => {
 accountAlice = f[1];
 accountBob = f[2];
 account = accountAlice;
})

abi = JSON.parse('[{"constant":false,"inputs":[],"name":"emptyShoppingList","outputs":[],"payable":false,"stateMutability":"nonpayable","type":"function"},{"constant":false,"inputs":[{"name":"shoppingListItem","type":"bytes32"}],"name":"addElementToShoppingList","outputs"

contract = new web3.eth.Contract(abi);
contract.options.address = "0x9Dfb46Fe1ecb1D1185B504e952d96e003C25e37C";
// update this contract address with your contract address

$("#selectUser").change(function(){
    if($(this).children("option:selected").val() == "accountBob"){
        account = accountBob;
    } else {
        account = accountAlice;
    }
    if($("#shoppingListTable").children().length != 0){
        $("#shoppingListTable").empty();
        //Delete all elements of the list when reload
        contract.methods.emptyShoppingList().send({from: account}).then((f) => console.log(f))
    }
});

//Method for adding elements to the table and executing them in the blockchain
function addElementToShoppingListJs(){
    var item = $("#item").val();
    //Add element to the HTML list
    $("#shoppingListTable").append("<tr>"+
     "<td>" + item + "</td>"+
    "</tr>");

    var itemHierarchyzed;

    //Adding the hierarchy
    switch (item){
        case "Banana":
        case "Apple":
        case "Mango":
            itemHierarchyzed = "Fruit";
            break;
        case "Sausage":
        case "Chicken":
        case "Hamburger":
        case "Bacon":
            itemHierarchyzed = "Meat";
            break;
        case "Salmon":
        case "Shrimp":
        case "Crab":
            itemHierarchyzed = "Fish";
            break;
        case "Coke":
        case "Beer":
        case "Water":
            itemHierarchyzed = "Drink";
            break;
        default:
            itemHierarchyzed = "Other";
    }


    //Add element to the blockchain
    contract.methods.addElementToShoppingList(web3.utils.asciiToHex(itemHierarchyzed)).send({from: account}).then((f) => console.log(f))
    $("#item").val('');
}

$(document).ready(function() {
    //Delete all elements of the list when reload
    contract.methods.emptyShoppingList().send({from: account}).then((f) => console.log(f))
    if($("#shoppingListTable").children().length != 0){
```

Line 54, Column 23 | Tab Size: 4 | JavaScript



```
pragma solidity >=0.4.0 <0.6.0;

contract shoppingListEasy{

  //Array for the list of items
  bytes32[] public shoppingList;

  //Constructor for initializing the contract with the shoppingList array
  constructor(bytes32[] memory shoppingListNames) public {
    shoppingList = shoppingListNames;
 }

 //Function for adding elements to the shopping list
 function addElementToShoppingList(bytes32 shoppingListItem){
   shoppingList.push(shoppingListItem);
 }
 //Function for obtaining the number of elements  in the shopping list
 function totalElementsShoppingList() view public returns (uint256) {
  return shoppingList.length;
 }
 //Function that returns all the elements of the shopping list
 function returnElementsShoppingList() view public returns (bytes32[]) {
   return shoppingList;
 }
 //Function that empty the shopping list
 function emptyShoppingList(){
   delete shoppingList;
 }

}
```