Etherium Blockchain Developer Technical documentation

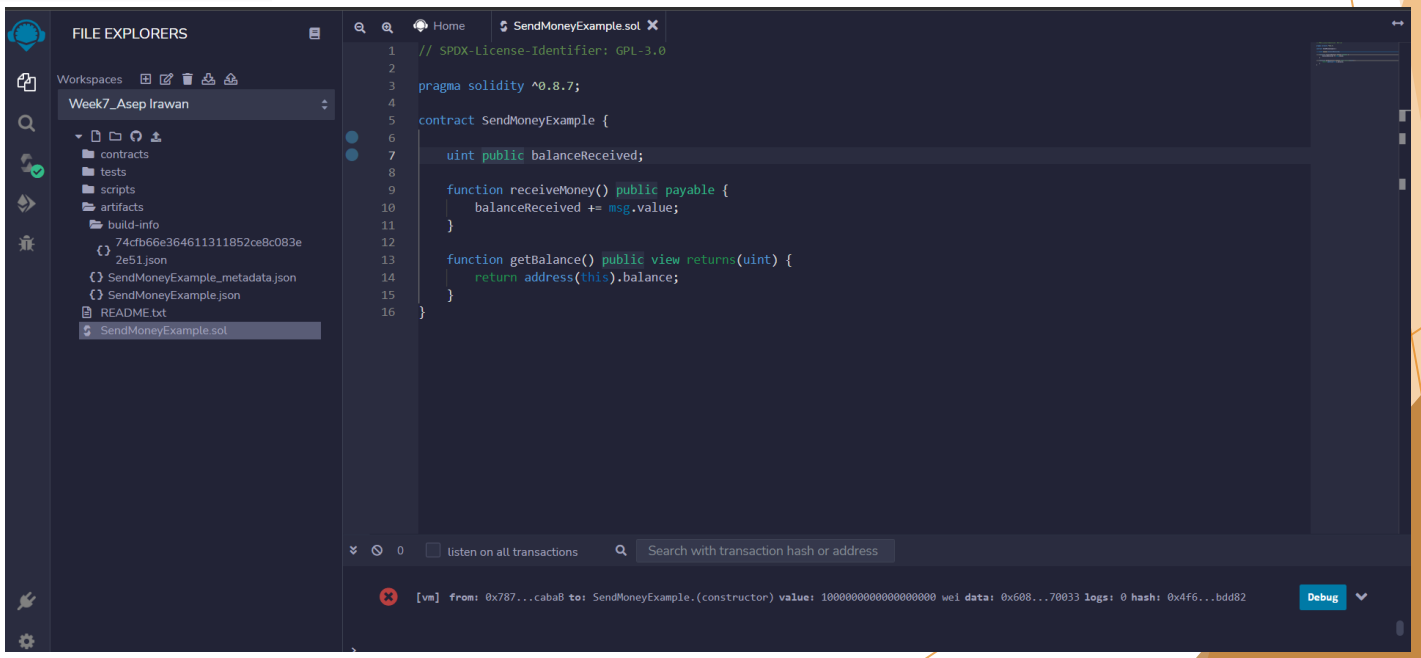
Nama : Asep Irawan

NIM : 1103190112

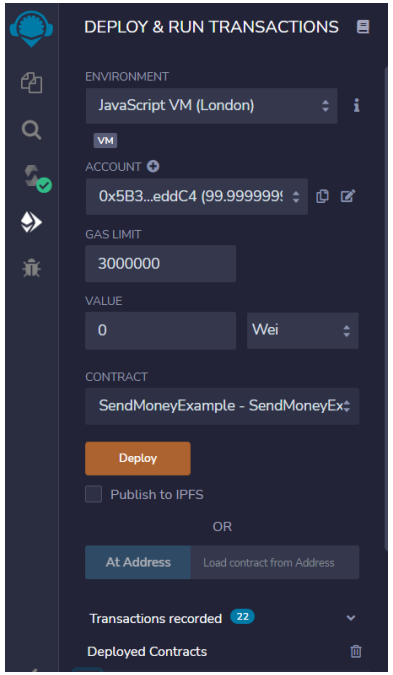
LAB : Deposit/Withdraw Ether

```
// SPDX-License-Identifier: GPL-3.0

pragma solidity ^0.8.1;

contract SendMoneyExample {

    uint public balanceReceived;

    function receiveMoney() public payable {
        balanceReceived += msg.value;
    }

    function getBalance() public view returns(uint) {
        return address(this).balance;
    }
}
```

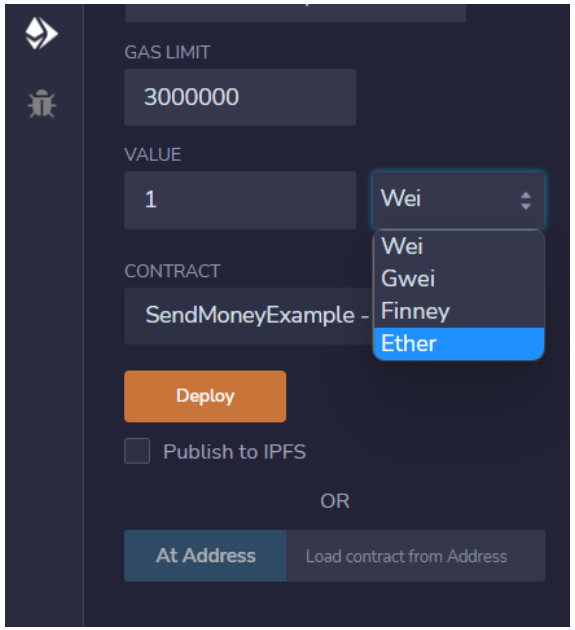
- Memasukan smart kontrak sederhana

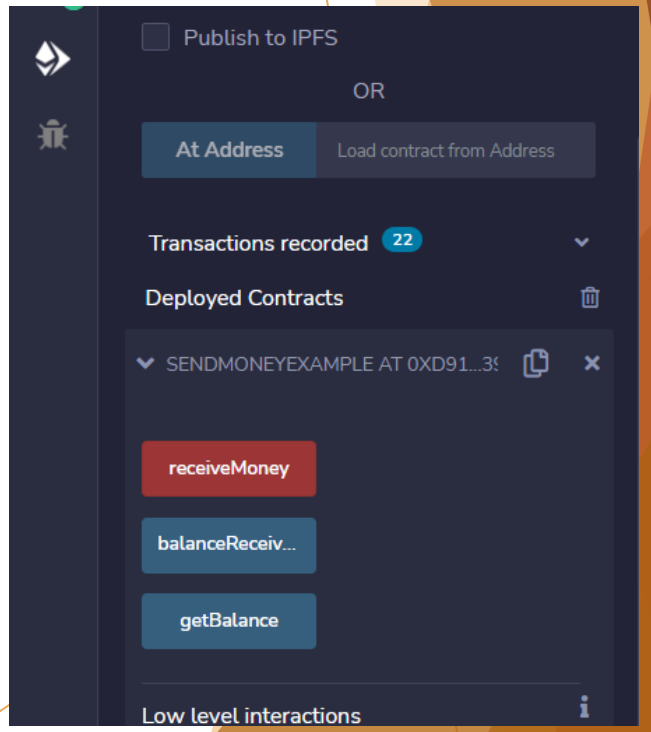


LAB : Deposit/Withdraw Ether

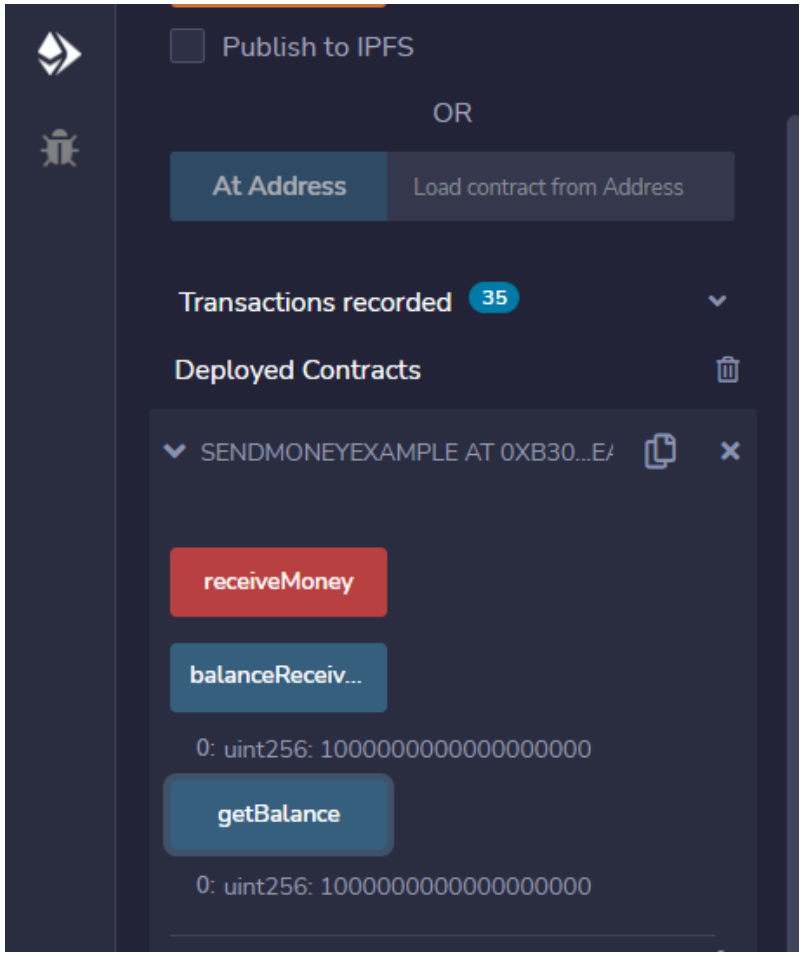


- Deploy Smart Contract
- Dan Mengganti Wei dengan ether
- Menekan ReceiveMoney





LAB : Deposit/Withdraw Ether



- Melakukan Pengecekan Balance

LAB : Deposit/Withdraw Ether

- Menambahkan fungsi Withdraw

Home | SendMoneyExample.sol

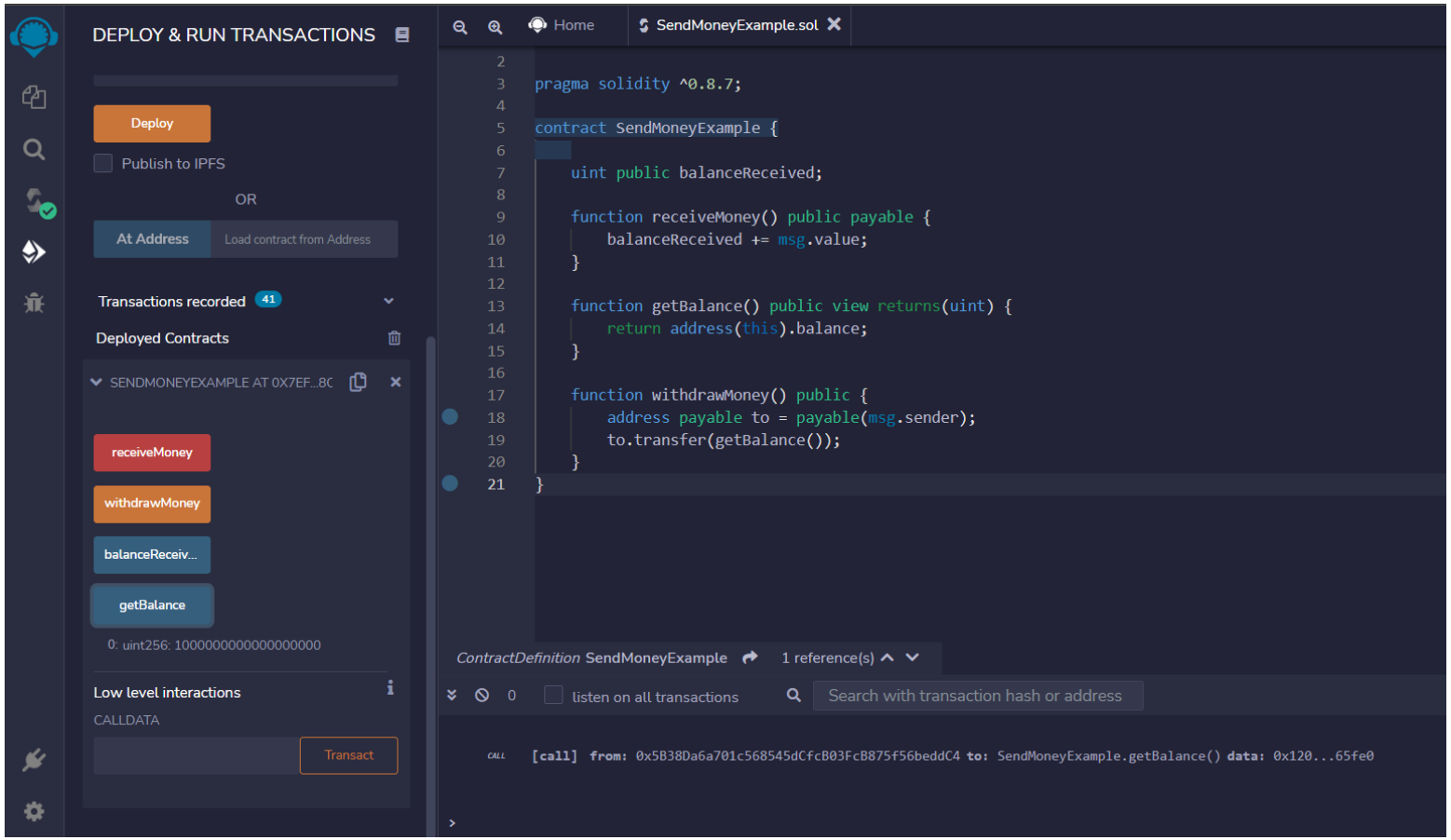
```
pragma solidity ^0.8.1;

contract SendMoneyExample {

uint public balanceReceived;
    uint public lockedUntil;

    function receiveMoney() public payable {
        balanceReceived += msg.value;
        lockedUntil = block.timestamp + 1 minutes;
    }
    function getBalance() public view returns(uint) {
        return address(this).balance;
    }
    function withdrawMoney() public {
        if(lockedUntil < block.timestamp) {
        address payable to = payable(msg.sender);
        to.transfer(getBalance());
        }
    }
    function withdrawMoneyTo(address payable _to) public {
        if(lockedUntil < block.timestamp) {
        _to.transfer(getBalance());
        }
    }
}
```

0 listen on all transactions Search with transaction hash or address

CALL [call] from: 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2 to: SendMoneyExample.getBalance() data: 0x120...65fe0

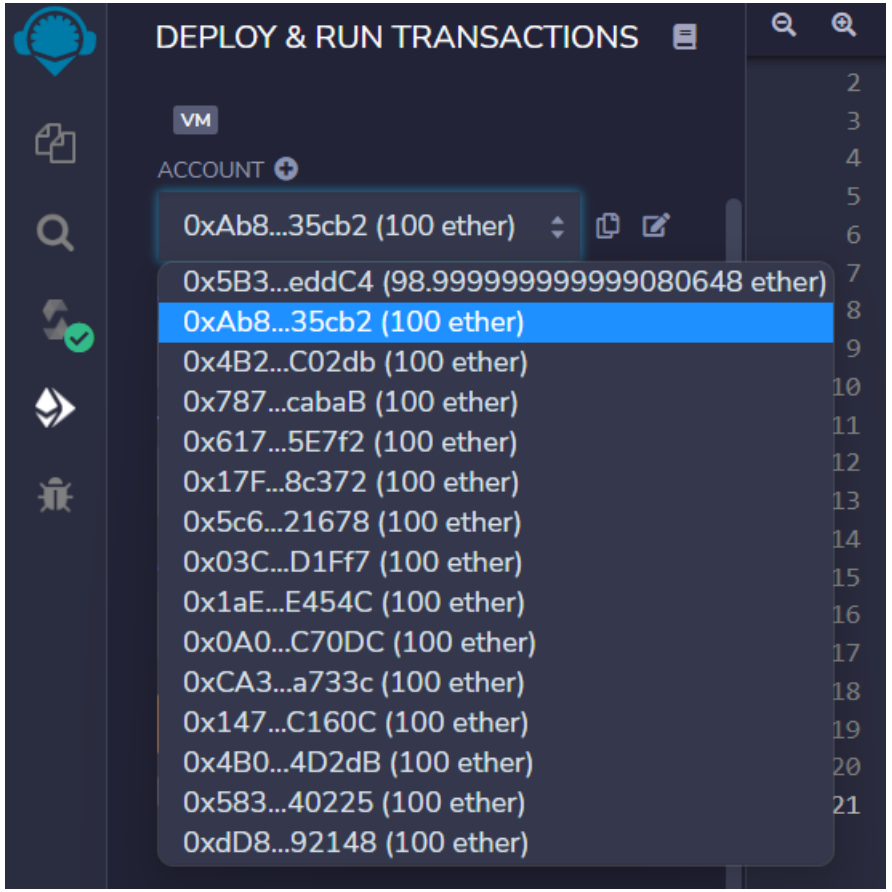
```
    function withdrawMoney() public {
        address payable to = payable(msg.sender);
        to.transfer(getBalance());
    }
}
```

LAB : Deposit/Withdraw Ether

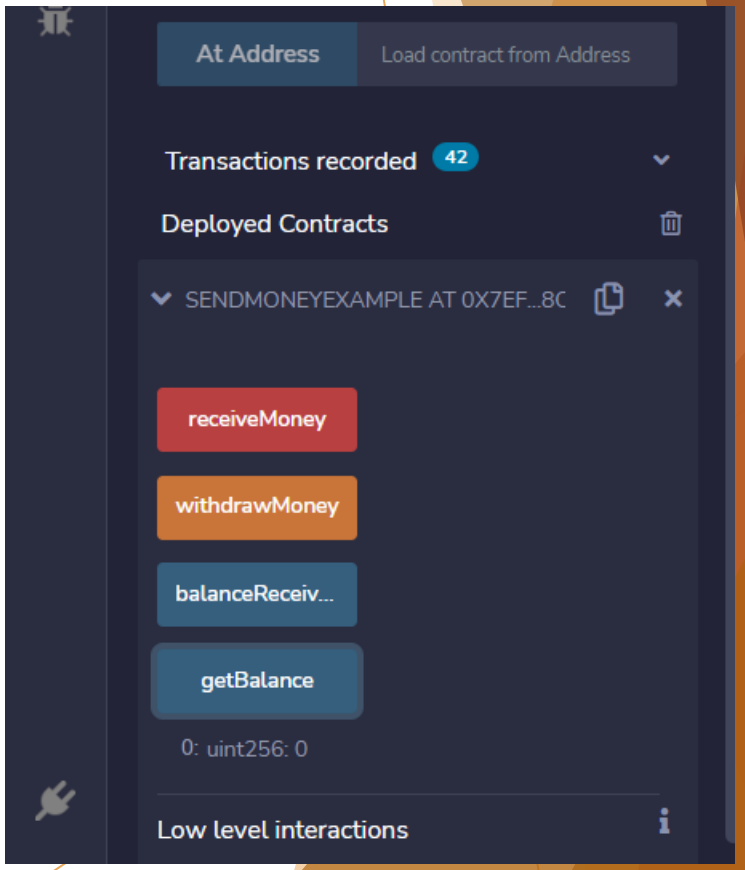


- ❑ Melakukan deploy ulang
- ❑ Melakukan receive money dengan 1 ether

LAB : Deposit/Withdraw Ether



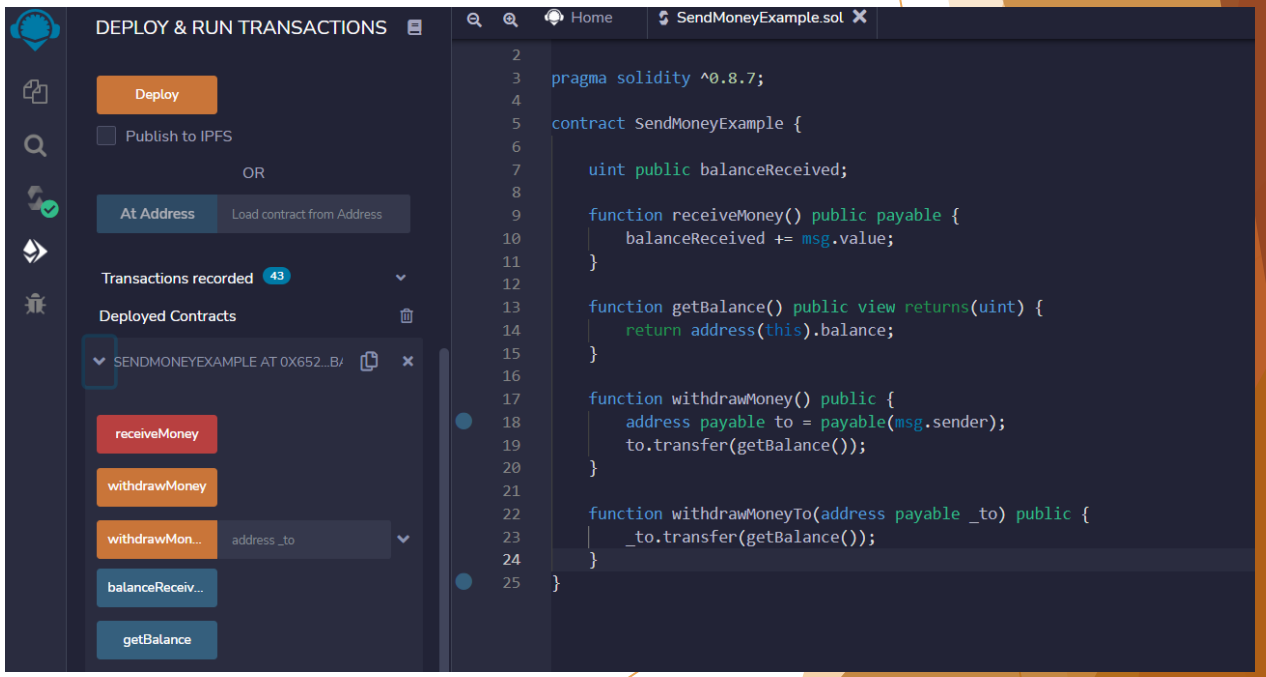
- Mengubah isi dari akun
- Melakukan Withdraw
- Mengecek Balance



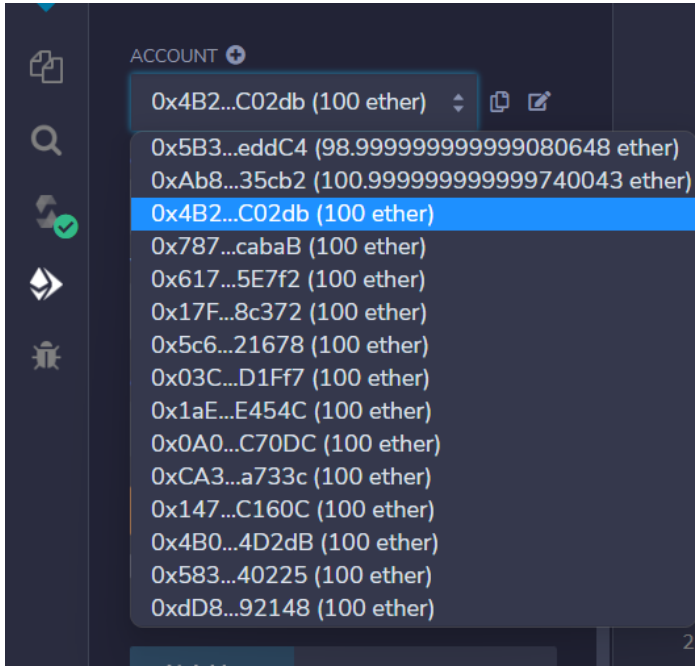
LAB : Deposit/Withdraw Ether

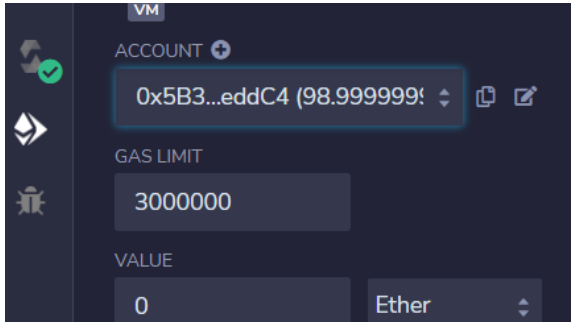
```
    function withdrawMoneyTo(address payable _to) public {
        _to.transfer(getBalance());
    }
}
```

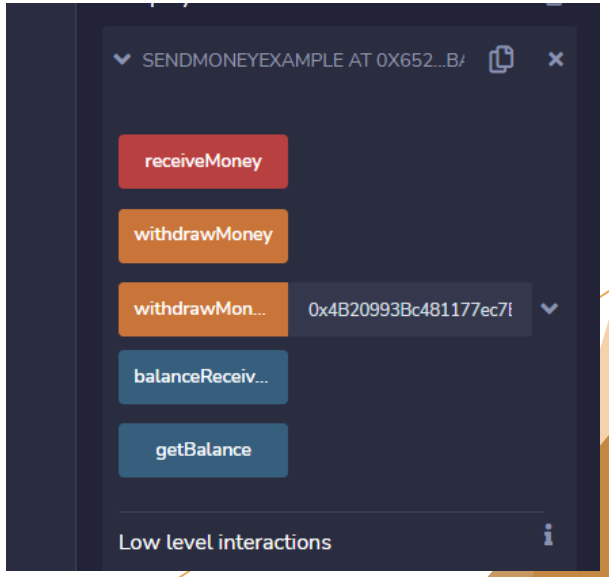
- Menambahkan Fungsi WithdrawMoneyTo



LAB : Deposit/Withdraw Ether

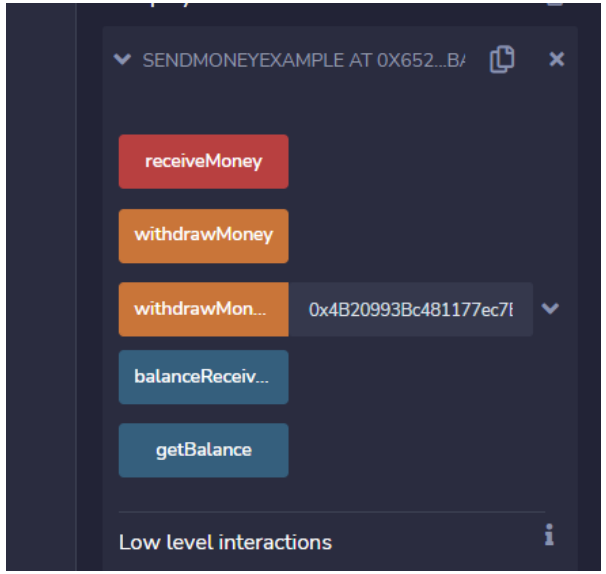


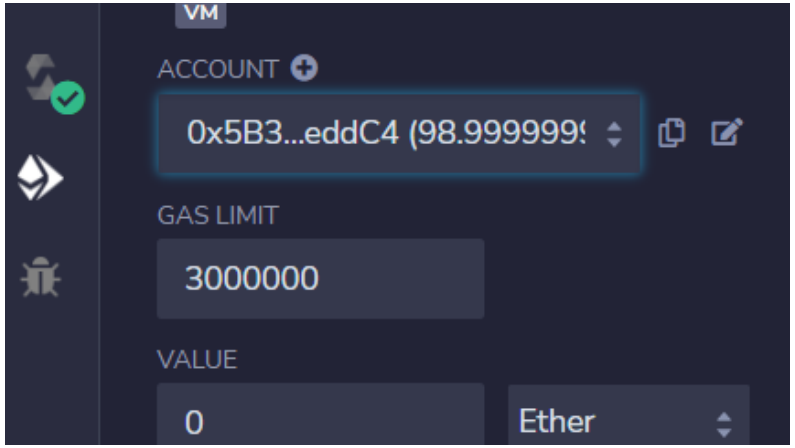




- Memilih Akun dan menyalin address
- Pindah pada akun yg sebelumnya
- Dan menempelkan pada withdrawnTo

LAB : Deposit/Withdraw Ether





- Setelah Menekan Withdrawn Money To
- Uang akan beralih pada address yang ditujukan sebelumnya

LAB : Deposit/Withdraw Ether

```
// SPDX-License-Identifier: GPL-3.0

pragma solidity ^0.8.1;

contract SendMoneyExample {

    uint public balanceReceived;
    uint public lockedUntil;

    function receiveMoney() public payable {
        balanceReceived += msg.value;
        lockedUntil = block.timestamp + 1 minutes;
    }

    function getBalance() public view returns(uint) {
        return address(this).balance;
    }

    function withdrawMoney() public {
        if(lockedUntil < block.timestamp) {
            address payable to = payable(msg.sender);
            to.transfer(getBalance());
        }
    }

    function withdrawMoneyTo(address payable _to) public {
        if(lockedUntil < block.timestamp) {
            _to.transfer(getBalance());
        }
    }
}
```

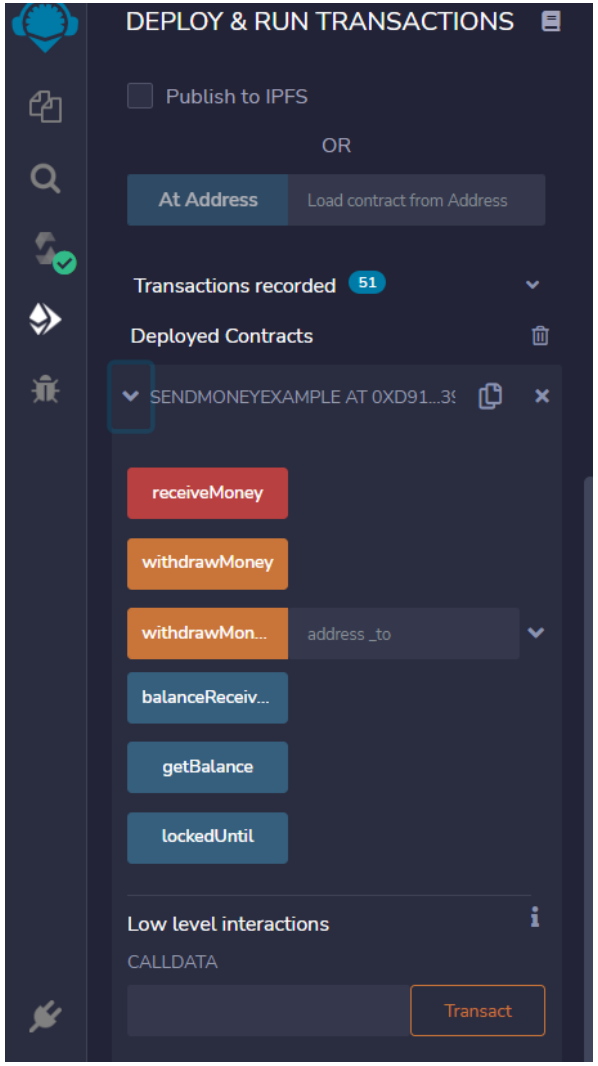
- Menambahkan Fungsi lock pada script

Home | README.txt | SendMoneyExample.sol

```
pragma solidity ^0.8.1;

contract SendMoneyExample {

uint public balanceReceived;
    uint public lockedUntil;

    function receiveMoney() public payable {
        balanceReceived += msg.value;
        lockedUntil = block.timestamp + 1 minutes;
    }
    function getBalance() public view returns(uint) {
        return address(this).balance;
    }
    function withdrawMoney() public {
        if(lockedUntil < block.timestamp) {
        address payable to = payable(msg.sender);
        to.transfer(getBalance());
        }
    }
    function withdrawMoneyTo(address payable _to) public {
        if(lockedUntil < block.timestamp) {
        _to.transfer(getBalance());
        }
    }
}
```

0 listen on all transactions Search with transaction hash or address

LAB : Deposit/Withdraw Ether



- Dengan begitu menambahkan satu buah fungsi

LAB : Shared Wallet

```
//SPDX-License-Identifier: MIT

pragma solidity 0.8.1;

contract SharedWallet {

    function withdrawMoney(address payable _to, uint _amount) public {
        _to.transfer(_amount);
    }

    receive() external payable {

    }
}
```

- Mendefinisikan dasar smart kontrak

Home | SharedWallet.sol

```
//SPDX-License-Identifier: MIT

pragma solidity 0.8.13;

contract SharedWallet {
    function withdrawMoney(address payable _to, uint _amount) public {
    _to.transfer(_amount);
    }
    receive() external payable {
    }
}
```

LAB : Shared Wallet

Home | SharedWallet.sol

```
//SPDX-License-Identifier: MIT

pragma solidity 0.8.13;

contract SharedWallet {
    address owner;
        constructor() {
        owner = msg.sender;
    }
    modifier onlyOwner() {
        require(msg.sender == owner, "You are not allowed");
        _;
    }
    function withdrawMoney(address payable _to, uint _amount) public onlyOwner {
        _to.transfer(_amount);
        }

    receive() external payable {
    }
}
```

- Membuat hanya pemilik yang dapat mengambilether

LAB : Shared Wallet

Home | SharedWallet.sol

```
//SPDX-License-Identifier: MIT

pragma solidity 0.8.13;

import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/access/Ownable.sol";

contract SharedWallet is Ownable {
   function isOwner() internal view returns(bool) {
        return owner() == msg.sender;
        }
    function withdrawMoney(address payable _to, uint _amount) public onlyOwner {
        _to.transfer(_amount);
        }

    receive() external payable {
    }
}
```

- Membuat ownable dengan menggunakan Smart kontrak dari OpenZeppelin

LAB : Shared Wallet

Home | SharedWallet.sol

```
import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/access/Ownable.s

contract SharedWallet is Ownable {
    function isOwner() internal view returns(bool) {
    return owner() == msg.sender;
    }

    mapping(address => uint) public allowance;

    function addAllowance(address _who, uint _amount) public onlyOwner {
        allowance[_who] = _amount;
    }

    modifier ownerOrAllowed(uint _amount) {

    require(isOwner() || allowance[msg.sender] >= _amount, "You are not allowed!");
        _;
    }

    function withdrawMoney(address payable _to, uint _amount) public ownerOrAllowed(_amount) {
        require(_amount <= address(this).balance, "Contract doesn't own enough money");
        _to.transfer(_amount);
    }

    receive() external payable {
    }
}
```

- Menambahkan penambahan Allowance dari sisi luar

LAB : Shared Wallet

Home | SharedWallet.sol

```
11
12      mapping(address => uint) public allowance;
13
14      function addAllowance(address _who, uint _amount) public onlyOwner {
15          allowance[_who] = _amount;
16      }
17
18      modifier ownerOrAllowed(uint _amount) {
19
20      require(isOwner() || allowance[msg.sender] >= _amount, "You are not allowed!");
21          _;
22      }
23
24  function reduceAllowance(address _who, uint _amount) internal ownerOrAllowed(_amount) {
25      allowance[_who] -= _amount;
26      }
27          function withdrawMoney(address payable _to, uint _amount) public ownerOrAllowed(_amount) {
28          require(_amount <= address(this).balance, "Contract doesn't own enough money");
29          if(!isOwner()) {
30              reduceAllowance(msg.sender, _amount);
31          }
32          _to.transfer(_amount);
33      }
34
35      receive() external payable {
36      }
37  }
```

- Memperbaiki Allowance untuk menghindari pengeluaran dua kali lipat

LAB : Shared Wallet

Home | Allowance.sol | SharedWallet.sol

```
//SPDX-License-Identifier: MIT

pragma solidity 0.8.1;

import "./Allowance.sol";

contract SharedWallet is Allowance {
    event MoneySent(address indexed _beneficiary, uint _amount);
    event MoneyReceived(address indexed _from, uint _amount);

    function withdrawMoney(address payable _to, uint _amount) public ownerOrAllowed(_amount) {
            require(_amount <= address(this).balance, "Contract doesn't own enough money");
            if(!isOwner()) {
                reduceAllowance(msg.sender, _amount);
            }
            emit MoneySent(_to, _amount);
            _to.transfer(_amount);
        }

        function renounceOwnership() public override onlyOwner {
            revert("can't renounceOwnership here"); //not possible with this smart contract
        }

        receive() external payable {
            emit MoneyReceived(msg.sender, msg.value);
        }
}
```

0 listen on all transactions Search with transaction hash or address

- Setelah melakukan penambahan script dan komposisi penyesuaian, kedua buah file dipisah
- Menjadi Allowance dan SharedWallet

Home | Allowance.sol | SharedWallet.sol

```
//SPDX-License-Identifier: MIT

pragma solidity 0.8.1;

import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/access/Ownable.sol";

contract Allowance is Ownable {
    event AllowanceChanged(address indexed _forWho, address indexed _byWhom, uint _oldAmount, uint _newAmount);
    mapping(address => uint) public allowance;

    function isOwner() internal view returns(bool) {
        return owner() == msg.sender;
    }

    function setAllowance(address _who, uint _amount) public onlyOwner {
        emit AllowanceChanged(_who, msg.sender, allowance[_who], _amount);
        allowance[_who] = _amount;
    }

    modifier ownerOrAllowed(uint _amount) {
        require(isOwner() || allowance[msg.sender] >= _amount, "You are not allowed!");
        _;
    }

    function reduceAllowance(address _who, uint _amount) internal ownerOrAllowed(_amount) {
        emit AllowanceChanged(_who, msg.sender, allowance[_who], allowance[_who] - _amount);
        allowance[_who] -= _amount;
    }
```

LAB :Suply Chain Project

```
pragma solidity ^0.6.0;
contract ItemManager{
    enum SupplyChainSteps{Created, Paid, Delivered}
    struct S_Item {
        ItemManager.SupplyChainSteps _step;
        string _identifier;
        uint _priceInWei;
    }
    mapping(uint => S_Item) public items;
    uint index;
    event SupplyChainStep(uint _itemIndex, uint _step);
    function createItem(string memory _identifier, uint _priceInWei) public {
        items[index]._priceInWei = _priceInWei;
        items[index]._step = SupplyChainSteps.Created;
        items[index]._identifier = _identifier;
        emit SupplyChainStep(index, uint(items[index]._step));
        index++;
    }
    function triggerPayment(uint _index) public payable {
        require(items[_index]._priceInWei <= msg.value, "Not fully paid");
        require(items[_index]._step == SupplyChainSteps.Created, "Item is further in the supply chain"
        items[_index]._step = SupplyChainSteps.Paid;
        emit SupplyChainStep(_index, uint(items[_index]._step));
    }
    function triggerDelivery(uint _index) public {
        require(items[_index]._step == SupplyChainSteps.Paid, "Item is further in the supply chain");
```

listen on network Search with transaction hash or address

- Membuat file “ItemManager” yang memungkinkan akan penambahan item dan melakukan pembayaran
- Lalu membuat item baru dengan nama “item”

```
pragma solidity ^0.6.0;
import "./ItemManager.sol";
contract Item {
    uint public priceInWei;
    uint public paidWei;
    uint public index;
    ItemManager parentContract;
    constructor(ItemManager _parentContract, uint _priceInWei, uint _index) public {
        priceInWei = _priceInWei;
        index = _index;
        parentContract = _parentContract;
    }
    receive() external payable {
        require(msg.value == priceInWei, "We don't support partial payments");
        require(paidWei == 0, "Item is already paid!");
        paidWei += msg.value;
        (bool success, ) = address(parentContract).call{value:msg.value}(abi.encodeWithSignature("trig
        require(success, "Delivery did not work");
    }
    fallback () external {
    }
}
```

LAB :Suply Chain Project

Home | SendMoney.sol | basicsmart.sol | ItemManager.sol | item.sol

```
    enum SupplyChainSteps {Created, Paid, Delivered}
    event SupplyChainStep(uint _itemIndex, uint _step, address _address);
    function createItem(string memory _identifier, uint _priceInWei) public {
        Item item = new Item(this, _priceInWei, index);
        items[index]._item = item;
        items[index]._step = SupplyChainSteps.Created;
        items[index]._identifier = _identifier;
        emit SupplyChainStep(index, uint(items[index]._step), address(item));
        index++;
    }
    function triggerPayment(uint _index) public payable {
        Item item = items[_index]._item;
        require(address(item) == msg.sender, "Only items are allowed to update themselves");
        require(item.priceInWei() == msg.value, "Not fully paid yet");
        require(items[_index]._step == SupplyChainSteps.Created, "Item is further in the supply chain"
        items[_index]._step = SupplyChainSteps.Paid;
        emit SupplyChainStep(_index, uint(items[_index]._step), address(item));
    }
    function triggerDelivery(uint _index) public {
        require(items[_index]._step == SupplyChainSteps.Paid, "Item is further in the supply chain");
        items[_index]._step = SupplyChainSteps.Delivered;
        emit SupplyChainStep(_index, uint(items[_index]._step), address(items[_index]._item));
    }
}
```

- Menambahkan smart kontrak pada ItemManager
- Lalu membuat file dengan nama Ownable

```
pragma solidity ^0.6.0;
    contract Ownable {
        address public _owner;
        constructor () internal {
            _owner = msg.sender;
        }
/**
* @dev Throws if called by any account other than the owner.
*/
    modifier onlyOwner() {
        require(isOwner(), "Ownable: caller is not the owner");
        _;
    }
/**
* @dev Returns true if the caller is the current owner.
*/
    function isOwner() public view returns (bool) {
        return (msg.sender == _owner);
    }
}
```

LAB :Suply Chain Project

```
pragma solidity ^0.8.1;
import "nyoba/Ownable.sol";
import "nyoba/Item.sol";
contract ItemManager is Ownable{
    struct S_Item {
        Item _item;
        ItemManager.SupplyChainSteps _step;
        string _identifier;
    }
    mapping(uint => S_Item) public items;
    uint index;
    enum SupplyChainSteps {Created, Paid, Delivered}
    event SupplyChainStep(uint _itemIndex, uint _step, address _address);
    function createItem(string memory _identifier, uint _priceInWei) public onlyOwner {
        Item item = new Item(this, _priceInWei, index);
        items[index]._item = item;
        items[index]._step = SupplyChainSteps.Created;
        items[index]._identifier = _identifier;
        emit SupplyChainStep(index, uint(items[index]._step), address(item));
        index++;
    }
    function triggerPayment(uint _index) public payable {
        Item item = items[_index]._item;
        require(address(item) == msg.sender, "Only items are allowed to update themselves");
        require(item.priceInWei() == msg.value, "Not fully paid yet");
        require(items[_index]._step == SupplyChainSteps.Created, "Item is further in the supply chain"
```

- Mengubah file ItemManager dengan menambahkan hanya pemilik yang dapat melakukan pengubahan pada data

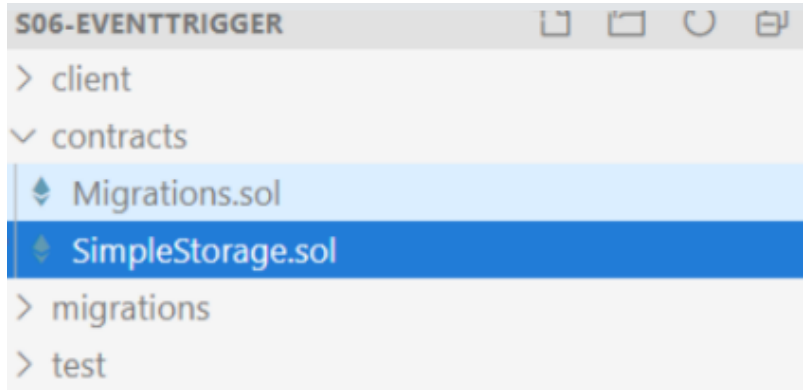
LAB :Suply Chain Project

```
    Directory: C:\101Tmp\ebd


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
d-----        1/11/2020  10:39 AM                s06-eventtrigger


ebd> cd .\s06-eventtrigger\
s06-eventtrigger> ls
```

- Setelah itu melakukan penginstalan Truffle dengan menggunakan Powershell
- Dan setelah selesai membuat file dengan peritah “ mkdir s06-eventtrigger cd s06-eventtrigger Ls”
- Lalu menjalankan “truffle unbox react”

```
    Directory: C:\101Tmp\ebd\s06-eventtrigger


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
d-----        1/11/2020  10:41 AM                client
d-----        1/11/2020  10:41 AM                contracts
d-----        1/11/2020  10:41 AM                migrations
d-----        1/11/2020  10:41 AM                test
-a----        1/11/2020  10:41 AM             33 .gitattributes
-a----        1/11/2020  10:41 AM           1075 LICENSE
-a----        1/11/2020  10:41 AM            297 truffle-config.js
```

LAB :Suply Chain Project



- Pada event trigger, hapus file “simpleStorage.sol”
- Sehingga file item, itemManager, dan ownable akan menambah sendiri

LAB :Suply Chain Project

```
const path = require("path");
module.exports = {
// See <http://truffleframework.com/docs/advanced/configuration>
// to customize your Truffle configuration!
contracts_build_directory: path.join(__dirname, "client/src/contracts"),
networks: {
develop: {
port: 8545
}
},
compilers: {
solc: {
version: "^0.6.0"
}
}
```

- Setelah mengatur compiler dari trufllenya
- Membuka powershell lagi dan melakukan perntah “Migrate”

```
truffle(develop)> migrate

Compiling your contracts...
===========================
> Compiling .\contracts\Item.sol
> Compiling .\contracts\ItemManager.sol
> Compiling .\contracts\Ownable.sol
> Artifacts written to C:\101Tmp\ebd\s06-eventtrigger\client\
> Compiled successfully using:
   - solc: 0.6.1+commit.e6f7d5a4.Emscripten.clang



Starting migrations...
```

LAB :Suply Chain Project

```
import React, { Component } from "react";
import ItemManager from "./contracts/ItemManager.json";
import Item from "./contracts/Item.json";
import getWeb3 from "./getWeb3";
import "./App.css";

class App extends Component {
  state = {cost: 0, itemName: "exampleItem1", loaded:false};

  componentDidMount = async () => {
    try {
      // Get network provider and web3 instance.
      this.web3 = await getWeb3();

      // Use web3 to get the user's accounts.
      this.accounts = await this.web3.eth.getAccounts();

      // Get the contract instance.
      const networkId = await this.web3.eth.net.getId();

      this.itemManager = new this.web3.eth.Contract(
        ItemManager.abi,
        ItemManager.networks[networkId] && ItemManager.networks[networkId].address,
      );
      this.item = new this.web3.eth.Contract(
        Item.abi,
        Item.networks[networkId] && Item.networks[networkId].address,
      );

      this.setState({loaded:true});

    } catch (error) {
      // Catch any errors for any of the above operations.
      alert(
        `Failed to load web3, accounts, or contract. Check console for details.`,
      );
      console.error(error);
    }
  };
//.. more code here ...
```

- Melakukan modifikasi pada file html
- Membuka file client/App.js dan memasukan script sebelumnya
- Lalu menambahkan render pada App.js
- Dan menambahkan fungsi handleInputChange dan handleSubmit

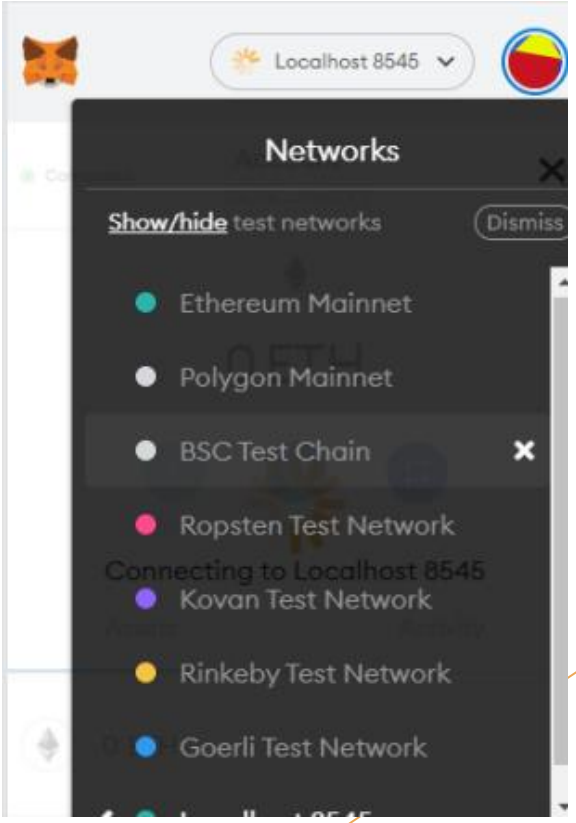
```
render() {
  if (!this.state.loaded) {
    return <div>Loading Web3, accounts, and contract...</div>;
  }
  return (
    <div class="App">
      <h1>Simply Payment/Supply Chain Example!</h1>
      <h2>Items</h2>

      <h2>Add Element</h2>
      Cost: <input type="text" name="cost" value={this.state.cost} onChange={this.handleInputChange} />
      Item Name: <input type="text" name="itemName" value={this.state.itemName} onChange={this.handleInputChange} />
      <button type="button" onClick={this.handleSubmit}>Create new Item</button>
    </div>
  );
}
```

```
handleSubmit = async () => {
    const { cost, itemName } = this.state;
    console.log(itemName, cost, this.itemManager);
    let result = await this.itemManager.methods.createItem(itemName, cost).send({ from: this.accounts[0] });
    console.log(result);
    alert("Send "+cost+" Wei to "+result.events.SupplyChainStep.returnValues._address);
  };

  handleInputChange = (event) => {
    const target = event.target;
    const value = target.type === 'checkbox' ? target.checked : target.value;
    const name = target.name;
```

LAB :Suply Chain Project

```
npm
Compiled successfully!

You can now view client in the browser.

  Local:            http://localhost:3000/
  On Your Network:  http://10.0.75.1:3000/

Note that the development build is not optimized.
To create a production build, use yarn build.
```

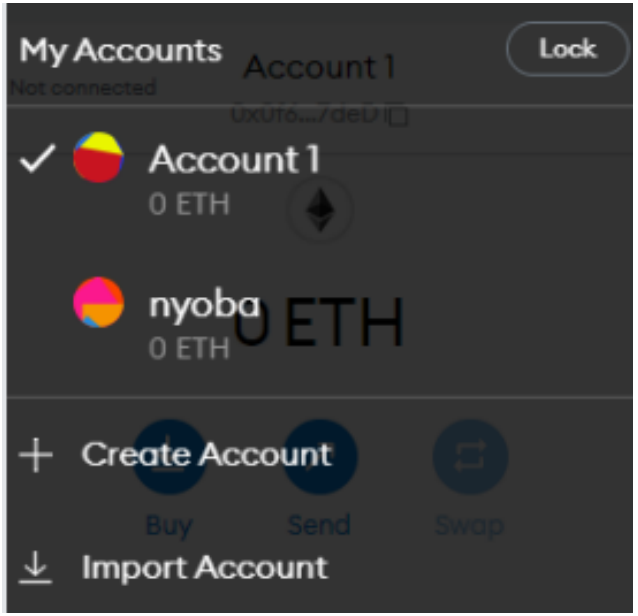
- Melakukan npm start pada powershell
- Dan beralih pada metamask untuk mengubah network menjadi “localhost 8545”



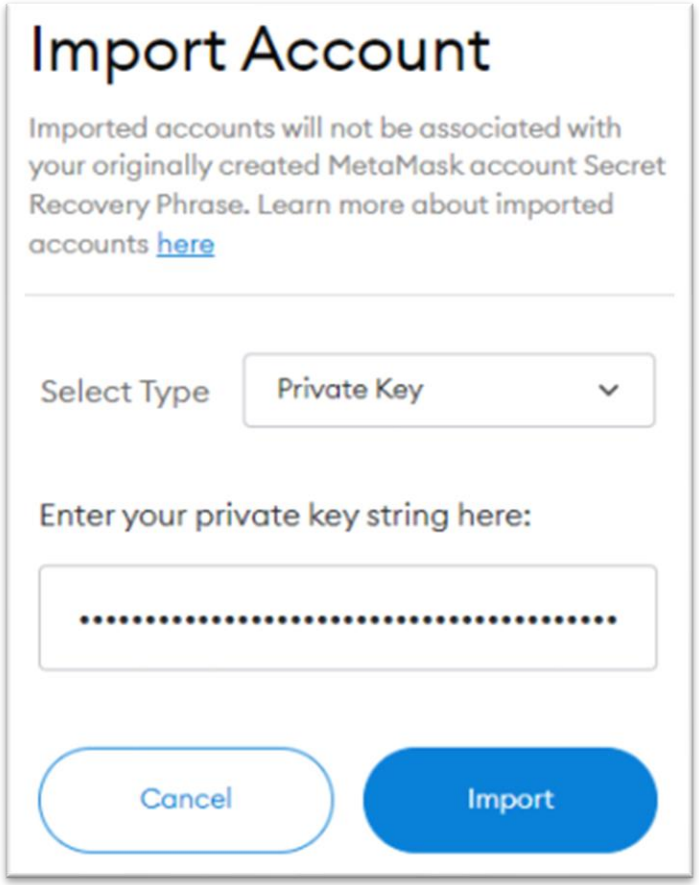
LAB :Suply Chain Project

- Setelah dilakukannya migrasi smart kontrak yang membuat truffle sebagai akun pertama dan pemilik
- Dan truffle pada powershell akan menghasilkan kunci
- Dan mengimport akun pada metamask

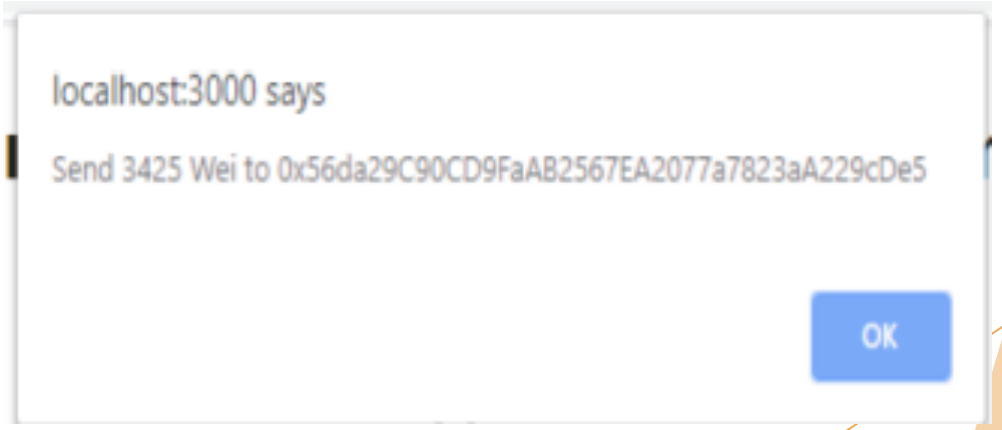
```
Private Keys:
(0) 2a9ed36cdb66f81093a82443c2b9f237f3534ef75f4f044fa6ebd76d5d05f
(1) f9c941a67e63fe4b84fe63ad652c29b2f225eb57562b246bf44bd3527b94b
```



LAB :Suply Chain Project



- Lalu Memsukan key yang didapat sebelumnya
- Dan akan diberikan notifikasi lebih lanjut



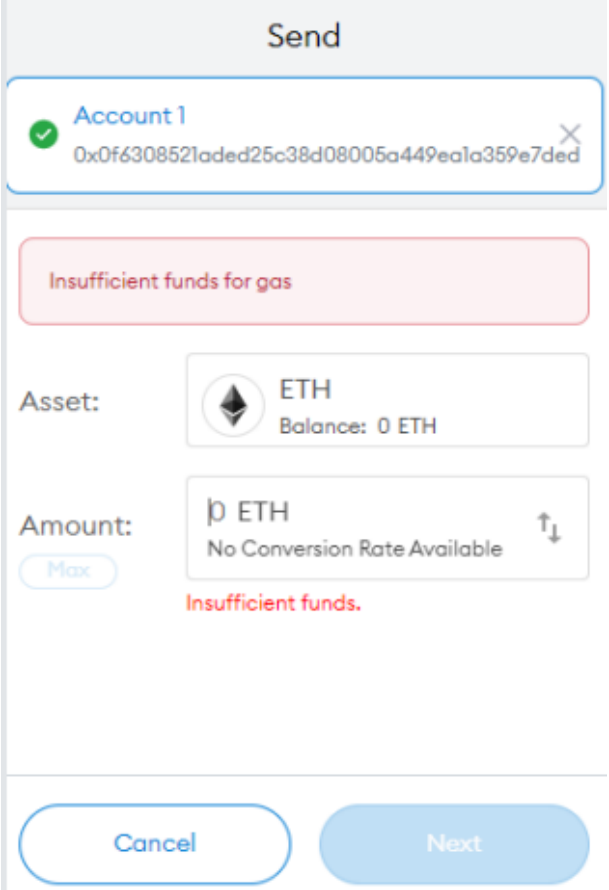
LAB :Suply Chain Project

- Menambahkan fungsi ListenToPayment pada App.js
- Dan memanggil fungsi ini dalam “componentDidMount”

```
listenToPaymentEvent = () => {
    let self = this;
    this.itemManager.events.SupplyChainStep().on("data", async function(evt) {
      if(evt.returnValues._step == 1) {
        let item = await self.itemManager.methods.items(evt.returnValues._itemIndex).call();
        console.log(item);
        alert("Item " + item._identifier + " was paid, deliver it now!");
      };
      console.log(evt);
    });
  }
```

```
//…
this.item = new this.web3.eth.Contract(
        ItemContract.abi,
        ItemContract.networks[this.networkId] && ItemContract.networks[this.networkId].address,
      );

      // Set web3, accounts, and contract to the state, and then proceed with an
      // example of interacting with the contract's methods.
      this.listenToPaymentEvent();
      this.setState({ loaded:true });
    } catch (error) {
      // Catch any errors for any of the above operations.
      alert(
        `Failed to load web3, accounts, or contract. Check console for details.`,
      );
      console.error(error);
    }
//...
```

LAB :Suply Chain Project



- Maka setelah itu memindahkan yang didapat dari truffle pada wallet

LAB :Suply Chain Project

```
const ItemManager = artifacts.require("./ItemManager.sol");

contract("ItemManager", accounts => {
it("... should let you create new Items.", async () => {
    const itemManagerInstance = await ItemManager.deployed();
    const itemName = "test1";
    const itemPrice = 500;

    const result = await itemManagerInstance.createItem(itemName, itemPrice, { from: accounts[0] });
    assert.equal(result.logs[0].args._itemIndex, 0, "There should be one item index in there")
    const item = await itemManagerInstance.items(0);
    assert.equal(item._identifier, itemName, "The item has a different identifier");
});
});
```

- Setelah membuat test kontrak
- Lalu memanggil “Truffle test” pada powershell

```
Compiling your contracts...
===========================
> Compiling .\contracts\Item.sol
> Compiling .\contracts\ItemManager.sol
> Compiling .\contracts\Ownable.sol



  Contract: ItemManager
    √ ... should let you create new Items. (160ms)


  1 passing (216ms)

s06-eventtrigger>
```