

Ethereum Blockchain Developer Technical documentation

Nama : Asep Irawan

NIM : 1103190112

LAB : Deposit/Withdraw Ether

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

pragma solidity ^0.8.1;

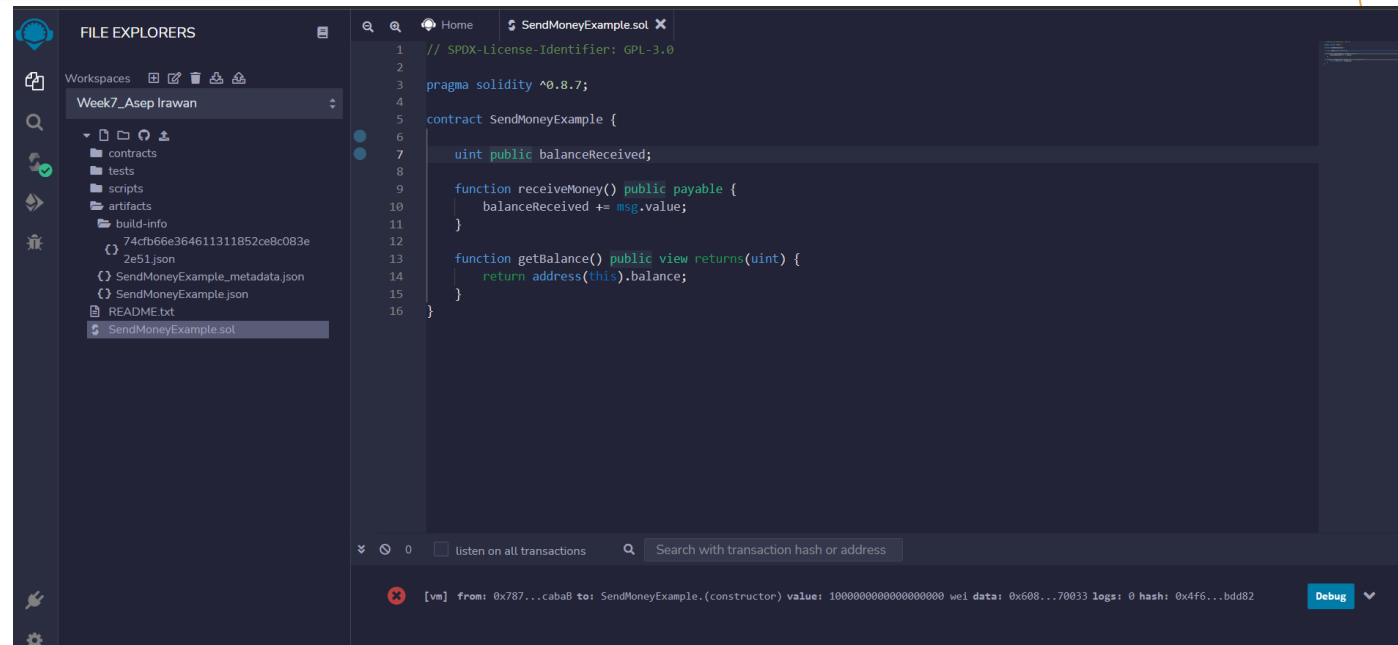
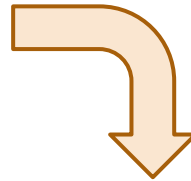
contract SendMoneyExample {

    uint public balanceReceived;

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

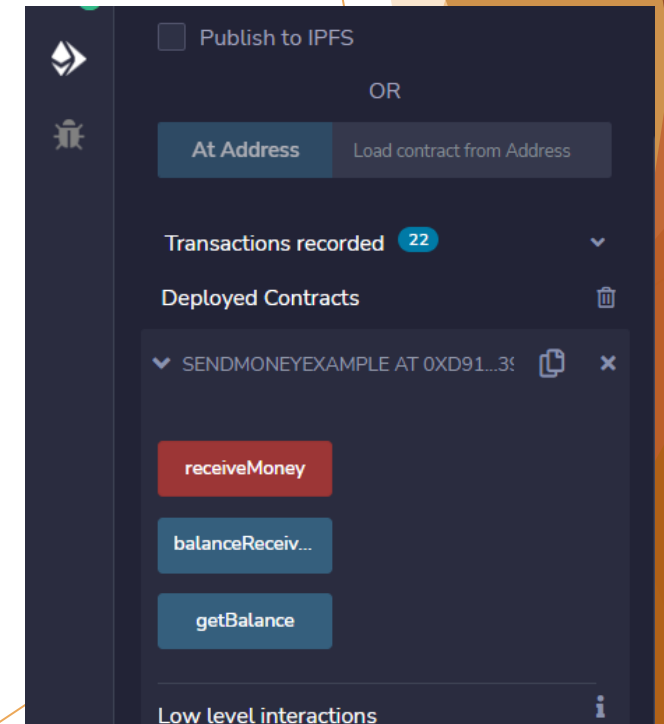
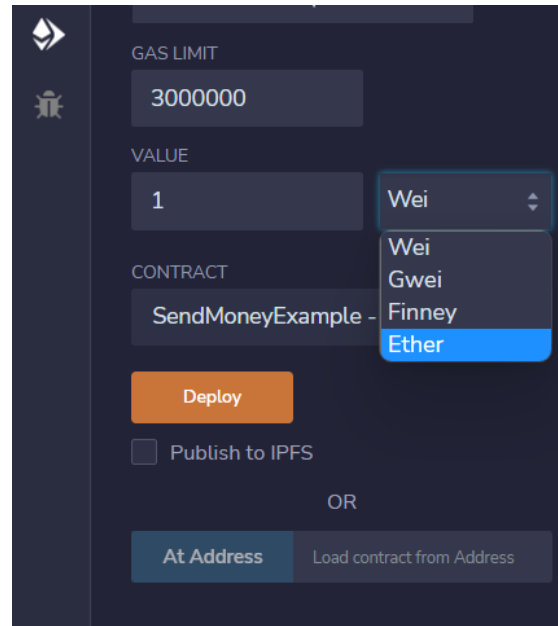
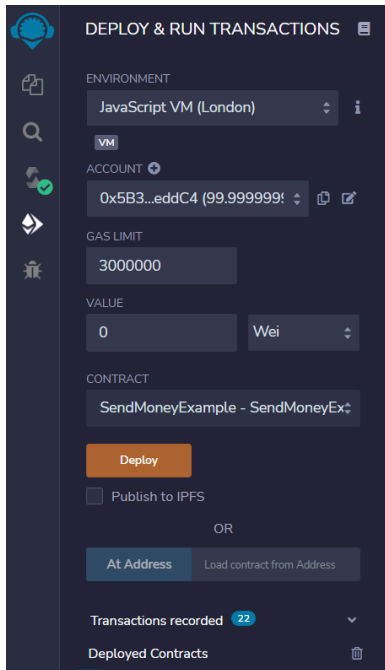
    function getBalance() public view returns(uint) {
        return address(this).balance;
    }
}
```

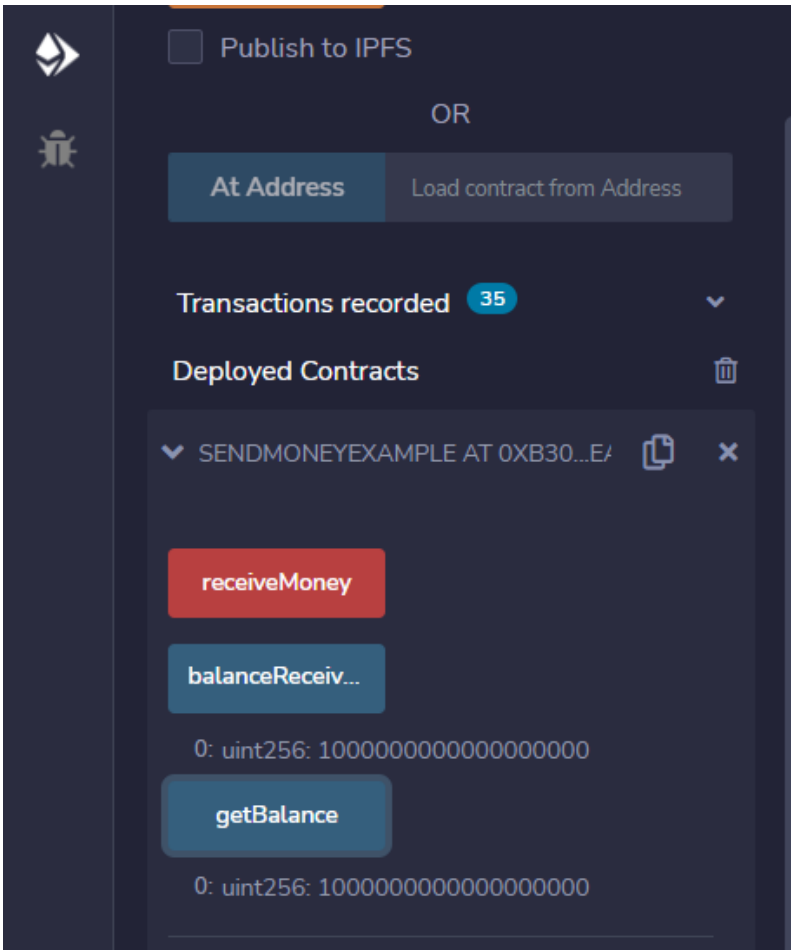
❑ Memasukan smart kontrak sederhana



LAB : Deposit/Withdraw Ether

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

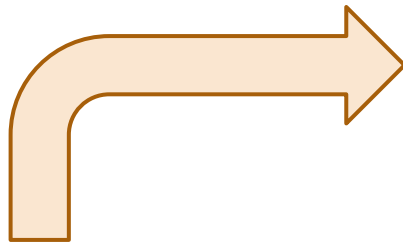




- ❑ Melakukan Pengecekan Balance

LAB : Deposit/Withdraw Ether

- ❑ Menambahkan fungsi Withdraw



```
2
3 pragma solidity ^0.8.1;
4
5 contract SendMoneyExample {
6
7     uint public balanceReceived;
8     uint public lockedUntil;
9
10    function receiveMoney() public payable {
11        balanceReceived += msg.value;
12        lockedUntil = block.timestamp + 1 minutes;
13    }
14    function getBalance() public view returns(uint) {
15        return address(this).balance;
16    }
17    function withdrawMoney() public {
18        if(lockedUntil < block.timestamp) {
19            address payable to = payable(msg.sender);
20            to.transfer(getBalance());
21        }
22    }
23    function withdrawMoneyTo(address payable _to) public {
24        if(lockedUntil < block.timestamp) {
25            _to.transfer(getBalance());
26        }
27    }
28 }
```

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

```
}
```

LAB : Deposit/Withdraw Ether

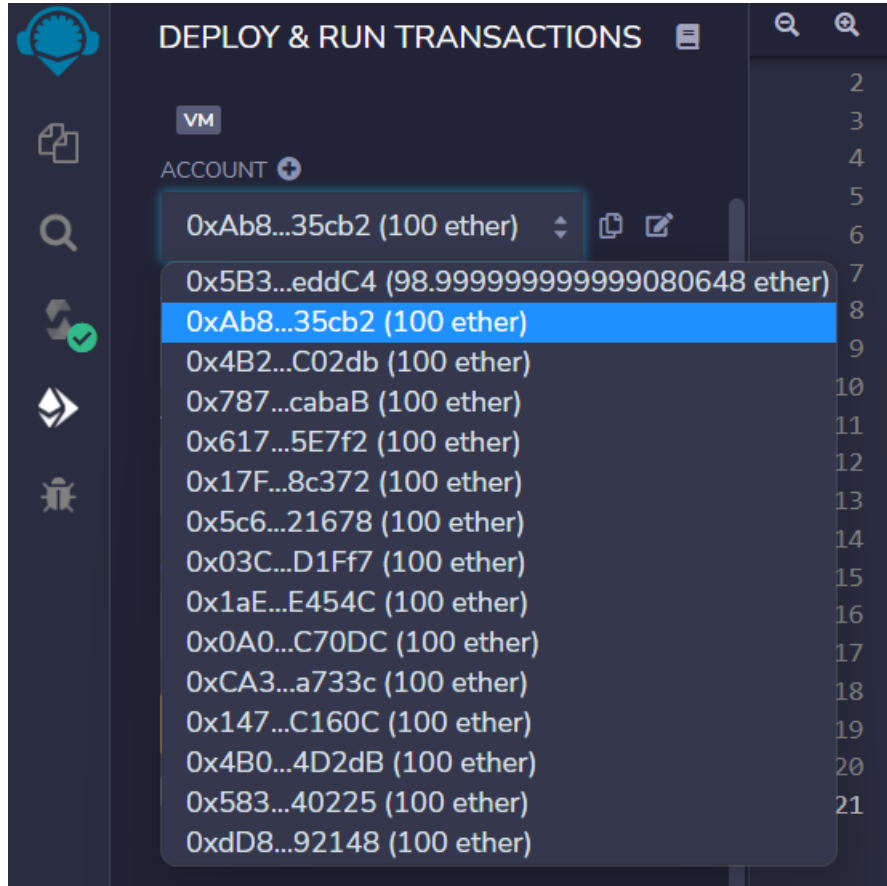
The screenshot displays the Remix IDE interface. On the left, the 'DEPLOY & RUN TRANSACTIONS' sidebar is active, showing a 'Deploy' button, a 'Publish to IPFS' checkbox, and a section for 'Transactions recorded' (41) and 'Deployed Contracts'. A contract named 'SENDMONEYEXAMPLE AT 0X7EF...8C' is selected, with buttons for 'receiveMoney', 'withdrawMoney', 'balanceReceiv...', and 'getBalance'. Below these, the 'Low level interactions' section shows a 'CALLDATA' field and a 'Transact' button. The main editor displays the Solidity code for 'SendMoneyExample.sol':

```
2  
3 pragma solidity ^0.8.7;  
4  
5 contract SendMoneyExample {  
6  
7     uint public balanceReceived;  
8  
9     function receiveMoney() public payable {  
10         balanceReceived += msg.value;  
11     }  
12  
13     function getBalance() public view returns(uint) {  
14         return address(this).balance;  
15     }  
16  
17     function withdrawMoney() public {  
18         address payable to = payable(msg.sender);  
19         to.transfer(getBalance());  
20     }  
21 }
```

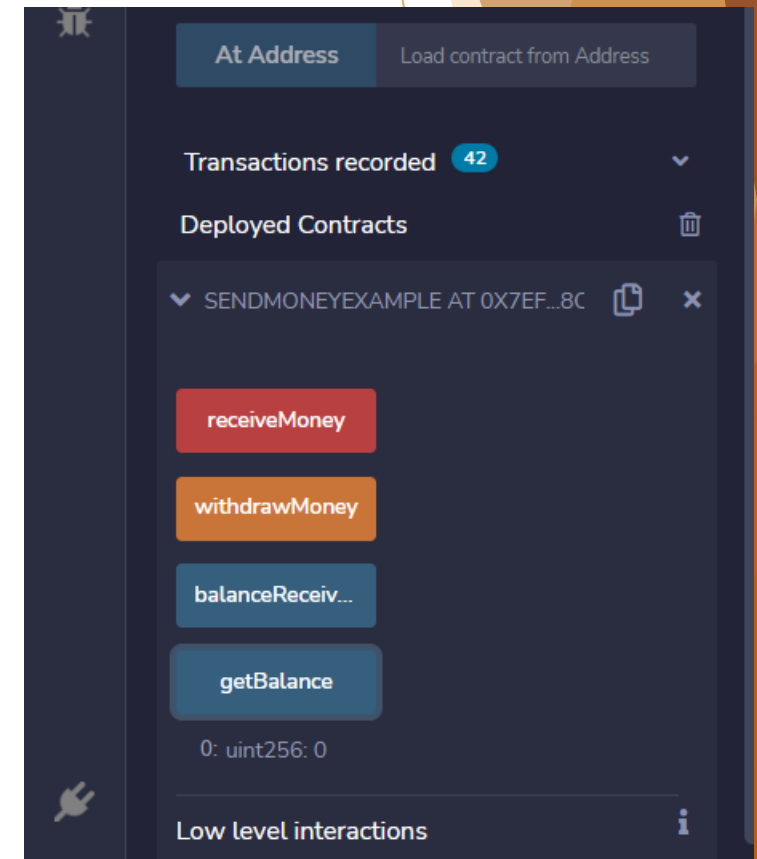
At the bottom, the 'ContractDefinition SendMoneyExample' section shows '1 reference(s)' and a search bar. Below this, a transaction log entry is visible: 'CALL [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: SendMoneyExample.getBalance() data: 0x120...65fe0'.

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

LAB : Deposit/Withdraw Ether



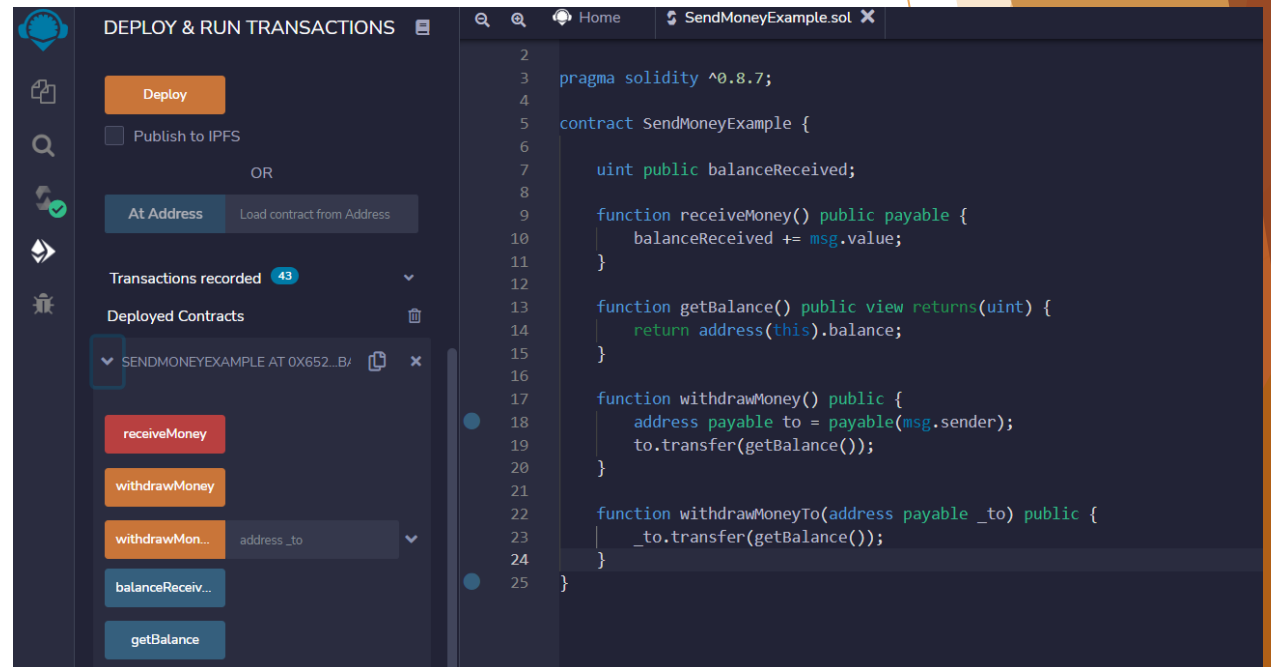
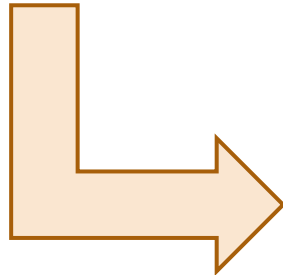
- ☐ Mengubah isi dari akun
- ☐ Melakukan Withdraw
- ☐ Mengecek Balance



LAB : Deposit/Withdraw Ether

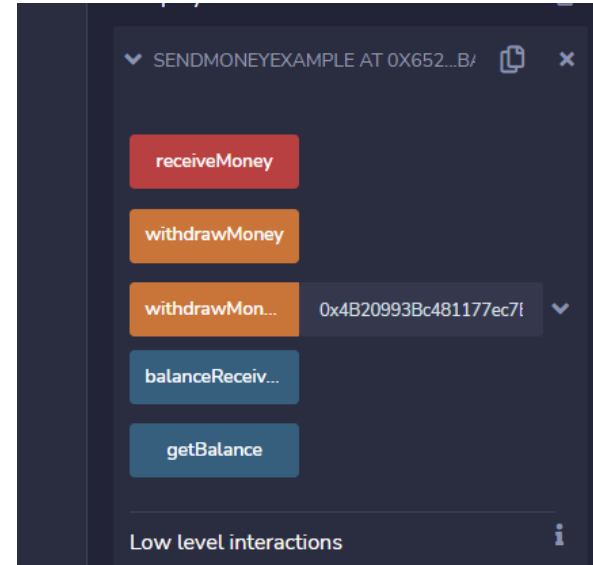
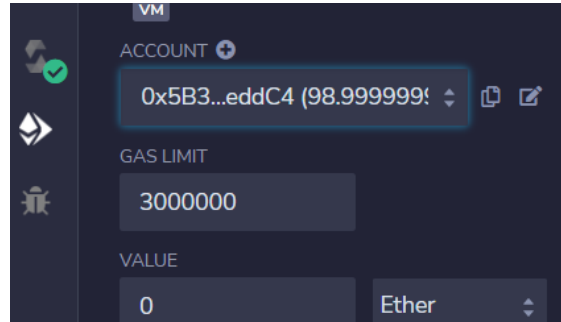
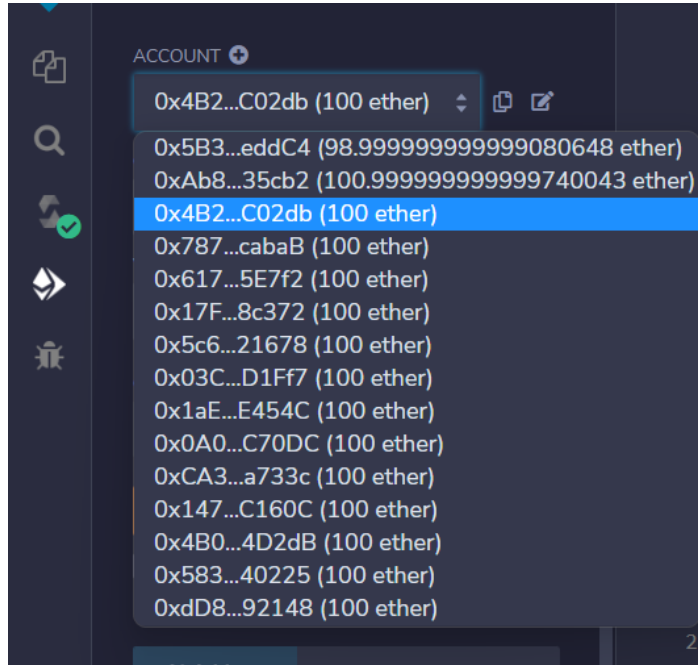
```
function withdrawMoneyTo(address payable _to) public {  
    _to.transfer(getBalance());  
}
```

- ❑ Menambahkan Fungsi WithdrawMoneyTo



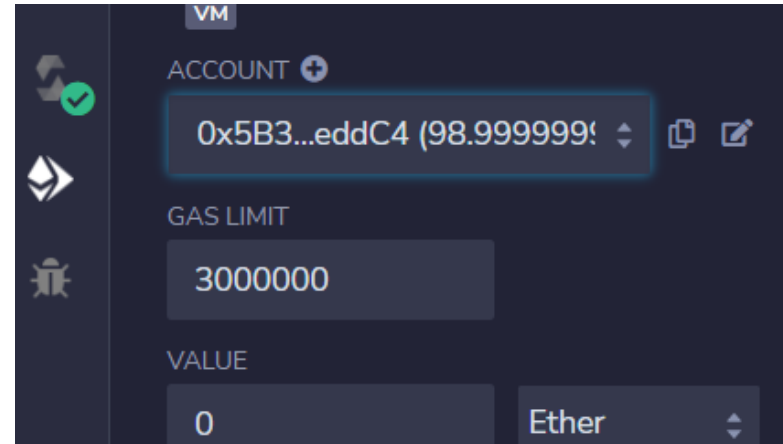
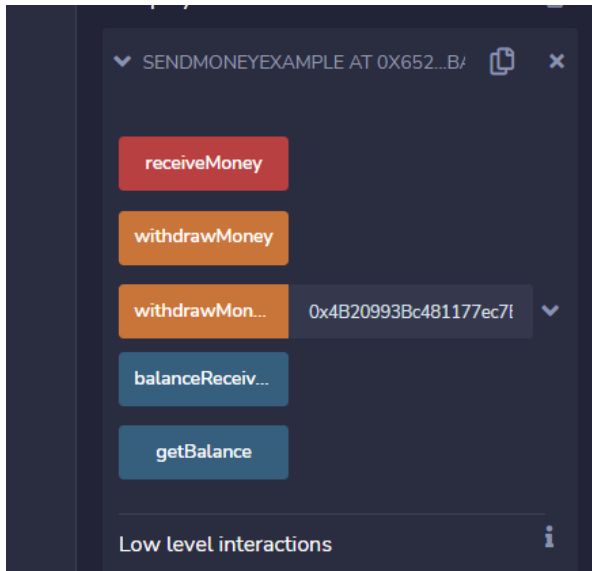
The screenshot displays the Remix IDE interface. On the left, the 'DEPLOY & RUN TRANSACTIONS' panel shows the 'Deploy' button, a 'Publish to IPFS' checkbox, and an 'At Address' button. Below this, it indicates 'Transactions recorded: 43' and 'Deployed Contracts: SENDMONEYEXAMPLE AT 0X652...B/'. A list of functions is shown: 'receiveMoney', 'withdrawMoney', 'withdrawMon...', 'balanceReceiv...', and 'getBalance'. On the right, the 'SendMoneyExample.sol' file is open, showing the Solidity code for the contract. The code includes a pragma statement, a contract definition with a public balance variable, a receiveMoney function, a getBalance function, a withdrawMoney function, and a withdrawMoneyTo function that transfers the contract's balance to a specified address.

```
2  
3 pragma solidity ^0.8.7;  
4  
5 contract SendMoneyExample {  
6  
7     uint public balanceReceived;  
8  
9     function receiveMoney() public payable {  
10         balanceReceived += msg.value;  
11     }  
12  
13     function getBalance() public view returns(uint) {  
14         return address(this).balance;  
15     }  
16  
17     function withdrawMoney() public {  
18         address payable to = payable(msg.sender);  
19         to.transfer(getBalance());  
20     }  
21  
22     function withdrawMoneyTo(address payable _to) public {  
23         _to.transfer(getBalance());  
24     }  
25 }
```

- ❑ 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;

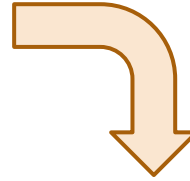
    function receiveMoney() public payable {
        balanceReceived += msg.value;
        lockedUntil = block.timestamp + 1 minutes;
    }

    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());
        }
    }

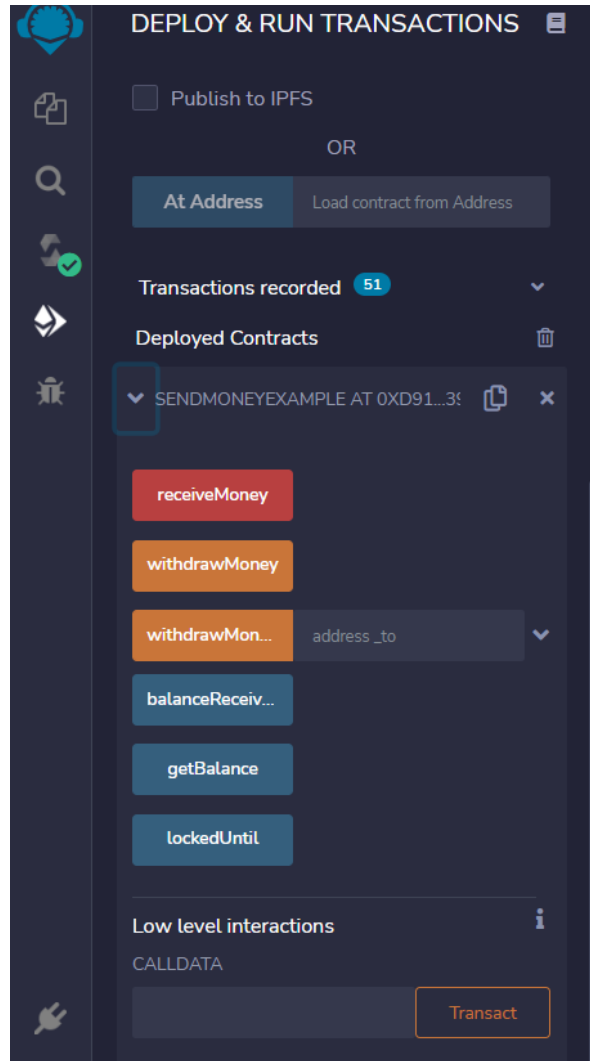
    function withdrawMoneyTo(address payable _to) public {
        if(lockedUntil < block.timestamp) {
            _to.transfer(getBalance());
        }
    }
}
```

❑ Menambahkan
Fungsi lock pada
script



```
2
3  pragma solidity ^0.8.1;
4
5  contract SendMoneyExample {
6
7      uint public balanceReceived;
8      uint public lockedUntil;
9
10     function receiveMoney() public payable {
11         balanceReceived += msg.value;
12         lockedUntil = block.timestamp + 1 minutes;
13     }
14     function getBalance() public view returns(uint) {
15         return address(this).balance;
16     }
17     function withdrawMoney() public {
18         if(lockedUntil < block.timestamp) {
19             address payable to = payable(msg.sender);
20             to.transfer(getBalance());
21         }
22     }
23     function withdrawMoneyTo(address payable _to) public {
24         if(lockedUntil < block.timestamp) {
25             _to.transfer(getBalance());
26         }
27     }
28 }
```

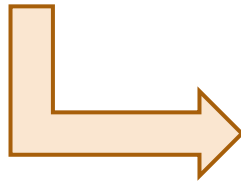
LAB : Deposit/Withdraw Ether



- ❑ 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 X  
1 //SPDX-License-Identifier: MIT  
2  
3 pragma solidity 0.8.13;  
4  
5 contract SharedWallet {  
6     function withdrawMoney(address payable _to, uint _amount) public {  
7         _to.transfer(_amount);  
8     }  
9     receive() external payable {  
10    }  
11 }
```

LAB : Shared Wallet

```
1 //SPDX-License-Identifier: MIT
2
3 pragma solidity 0.8.13;
4
5 contract SharedWallet {
6     address owner;
7     constructor() {
8         owner = msg.sender;
9     }
10    modifier onlyOwner() {
11        require(msg.sender == owner, "You are not allowed");
12        _;
13    }
14    function withdrawMoney(address payable _to, uint _amount) public onlyOwner {
15        _to.transfer(_amount);
16    }
17
18    receive() external payable {
19    }
20 }
```

- ❑ Membuat hanya pemilik yang dapat mengambilether



LAB : Shared Wallet

```
1 //SPDX-License-Identifier: MIT
2
3 pragma solidity 0.8.13;
4
5 import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/access/Ownable.sol";
6
7 contract SharedWallet is Ownable {
8     function isOwner() internal view returns(bool) {
9         return owner() == msg.sender;
10    }
11    function withdrawMoney(address payable _to, uint _amount) public onlyOwner {
12        _to.transfer(_amount);
13    }
14
15    receive() external payable {}
16
17 }
```

- ❑ Membuat ownable dengan menggunakan Smart kontrak dari OpenZeppelin



LAB : Shared Wallet

```
4
5 import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/access/Ownable.s
6
7 contract SharedWallet is Ownable {
8     function isOwner() internal view returns(bool) {
9         return owner() == msg.sender;
10    }
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 withdrawMoney(address payable _to, uint _amount) public ownerOrAllowed(_amount) {
25        require(_amount <= address(this).balance, "Contract doesn't own enough money");
26        _to.transfer(_amount);
27    }
28
29    receive() external payable {
30    }
31 }
```

- ❑ Menambahkan penambahan Allowance dari sisi luar

LAB : Shared Wallet

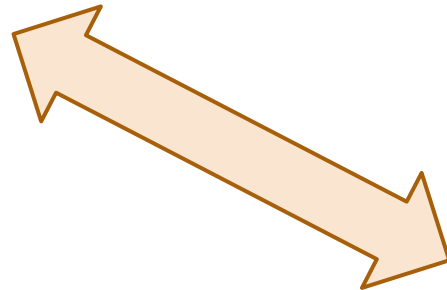
```
11 mapping(address => uint) public allowance;
12
13 function addAllowance(address _who, uint _amount) public onlyOwner {
14     allowance[_who] = _amount;
15 }
16
17 modifier ownerOrAllowed(uint _amount) {
18     require(isOwner() || allowance[msg.sender] >= _amount, "You are not allowed!");
19 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

- ❑ Memperbaiki Allowance untuk menghindari pengeluaran dua kali lipat

LAB : Shared Wallet

```
1 //SPDX-License-Identifier: MIT
2
3 pragma solidity 0.8.1;
4
5 import "../Allowance.sol";
6
7 contract SharedWallet is Allowance {
8     event MoneySent(address indexed _beneficiary, uint _amount);
9     event MoneyReceived(address indexed _from, uint _amount);
10
11     function withdrawMoney(address payable _to, uint _amount) public ownerOrAllowed(_amount) {
12         require(_amount <= address(this).balance, "Contract doesn't own enough money");
13         if(!isOwner()) {
14             reduceAllowance(msg.sender, _amount);
15         }
16         emit MoneySent(_to, _amount);
17         _to.transfer(_amount);
18     }
19
20     function renounceOwnership() public override onlyOwner {
21         revert("can't renounceOwnership here"); //not possible with this smart contract
22     }
23
24     receive() external payable {
25         emit MoneyReceived(msg.sender, msg.value);
26     }
27 }
```

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



```
1 //SPDX-License-Identifier: MIT
2
3 pragma solidity 0.8.1;
4
5 import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/access/Ownable.sol";
6
7 contract Allowance is Ownable {
8     event AllowanceChanged(address indexed _forWho, address indexed _byWhom, uint _oldAmount, uint _newAmount);
9     mapping(address => uint) public allowance;
10
11     function isOwner() internal view returns(bool) {
12         return owner() == msg.sender;
13     }
14
15     function setAllowance(address _who, uint _amount) public onlyOwner {
16         emit AllowanceChanged(_who, msg.sender, allowance[_who], _amount);
17         allowance[_who] = _amount;
18     }
19
20     modifier ownerOrAllowed(uint _amount) {
21         require(isOwner() || allowance[msg.sender] >= _amount, "You are not allowed!");
22         _;
23     }
24
25     function reduceAllowance(address _who, uint _amount) internal ownerOrAllowed(_amount) {
26         emit AllowanceChanged(_who, msg.sender, allowance[_who], allowance[_who] - _amount);
27         allowance[_who] -= _amount;
28     }
29 }
```

LAB : Suply Chain Project

```
1 pragma solidity ^0.6.0;
2 contract ItemManager{
3     enum SupplyChainSteps{Created, Paid, Delivered}
4     struct S_Item {
5         ItemManager.SupplyChainSteps _step;
6         string _identifier;
7         uint _priceInWei;
8     }
9     mapping(uint => S_Item) public items;
10    uint index;
11    event SupplyChainStep(uint _itemIndex, uint _step);
12    function createItem(string memory _identifier, uint _priceInWei) public {
13        items[index]._priceInWei = _priceInWei;
14        items[index]._step = SupplyChainSteps.Created;
15        items[index]._identifier = _identifier;
16        emit SupplyChainStep(index, uint(items[index]._step));
17        index++;
18    }
19    function triggerPayment(uint _index) public payable {
20        require(items[_index]._priceInWei <= msg.value, "Not fully paid");
21        require(items[_index]._step == SupplyChainSteps.Created, "Item is further in the supply chain");
22        items[_index]._step = SupplyChainSteps.Paid;
23        emit SupplyChainStep(_index, uint(items[_index]._step));
24    }
25    function triggerDelivery(uint _index) public {
26        require(items[_index]._step == SupplyChainSteps.Paid, "Item is further in the supply chain");
27        items[_index]._step = SupplyChainSteps.Delivered;
28    }
29 }
```

- ❑ 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("triggerPayment", index));
        require(success, "Delivery did not work");
    }
    fallback () external {
    }
}
```

LAB :Supply Chain Project

```
11 enum SupplyChainSteps {Created, Paid, Delivered}
12 event SupplyChainStep(uint _itemIndex, uint _step, address _address);
13 function createItem(string memory _identifier, uint _priceInWei) public {
14     Item item = new Item(this, _priceInWei, index);
15     items[index]._item = item;
16     items[index]._step = SupplyChainSteps.Created;
17     items[index]._identifier = _identifier;
18     emit SupplyChainStep(index, uint(items[index]._step), address(item));
19     index++;
20 }
21 function triggerPayment(uint _index) public payable {
22     Item item = items[_index]._item;
23     require(address(item) == msg.sender, "Only items are allowed to update themselves");
24     require(item.priceInWei() == msg.value, "Not fully paid yet");
25     require(items[_index]._step == SupplyChainSteps.Created, "Item is further in the supply chain");
26     items[_index]._step = SupplyChainSteps.Paid;
27     emit SupplyChainStep(_index, uint(items[_index]._step), address(item));
28 }
29 function triggerDelivery(uint _index) public {
30     require(items[_index]._step == SupplyChainSteps.Paid, "Item is further in the supply chain");
31     items[_index]._step = SupplyChainSteps.Delivered;
32     emit SupplyChainStep(_index, uint(items[_index]._step), address(items[_index]._item));
33 }
34 }
```



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

```
1 pragma solidity ^0.6.0;
2 contract Ownable {
3     address public _owner;
4     constructor () internal {
5         _owner = msg.sender;
6     }
7
8     /**
9      * @dev Throws if called by any account other than the owner.
10     */
11     modifier onlyOwner() {
12         require(isOwner(), "Ownable: caller is not the owner");
13         _;
14     }
15
16     /**
17      * @dev Returns true if the caller is the current owner.
18     */
19     function isOwner() public view returns (bool) {
20         return (msg.sender == _owner);
21     }
22 }
```

LAB :Supply Chain Project

```
1 pragma solidity ^0.8.1;
2 import "nyoba/Ownable.sol";
3 import "nyoba/Item.sol";
4 contract ItemManager is Ownable{
5     struct S_Item {
6         Item _item;
7         ItemManager.SupplyChainSteps _step;
8         string _identifier;
9     }
10    mapping(uint => S_Item) public items;
11    uint index;
12    enum SupplyChainSteps {Created, Paid, Delivered};
13    event SupplyChainStep(uint _itemIndex, uint _step, address _address);
14    function createItem(string memory _identifier, uint _priceInWei) public onlyOwner {
15        Item item = new Item(this, _priceInWei, index);
16        items[index]._item = item;
17        items[index]._step = SupplyChainSteps.Created;
18        items[index]._identifier = _identifier;
19        emit SupplyChainStep(index, uint(items[index]._step), address(item));
20        index++;
21    }
22    function triggerPayment(uint _index) public payable {
23        Item item = items[_index]._item;
24        require(address(item) == msg.sender, "Only items are allowed to update themselves");
25        require(item.priceInWei() == msg.value, "Not fully paid yet");
26        require(items[_index]._step == SupplyChainSteps.Created, "Item is further in the supply chain")
```

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

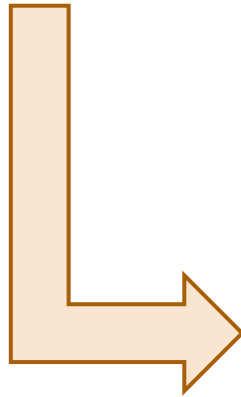
LAB :Supply Chain Project

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

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

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

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

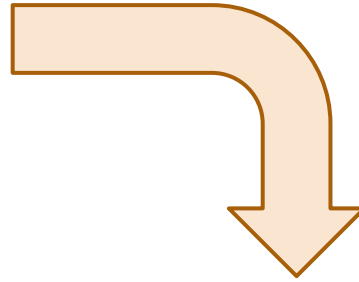
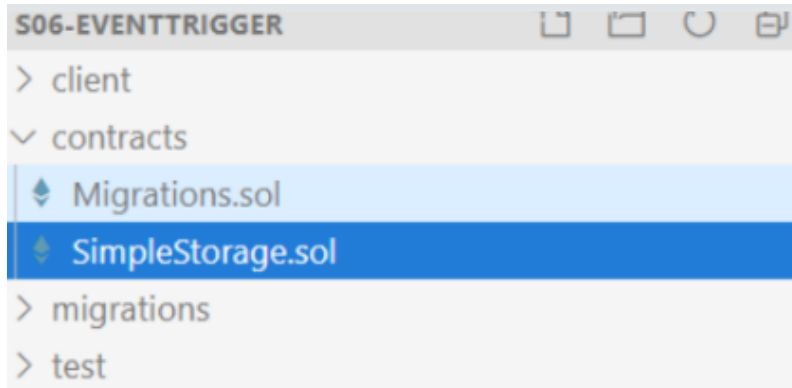


```
Directory: C:\101Tmp\ebd\s06-eventtrigger

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

s06-eventtrigger>
```

LAB :Supply Chain Project



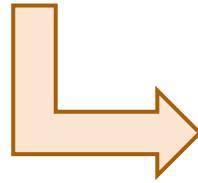
- ☐ Pada event trigger, hapus file “simpleStorage.sol”
- ☐ Sehingga file item, itemManager, dan ownable akan menambah sendiri



LAB :Supply Chain Project

```
1  const path = require("path");
2  module.exports = {
3    // See <http://truffleframework.com/docs/advanced/configuration>
4    // to customize your Truffle configuration!
5    contracts_build_directory: path.join(__dirname, "client/src/contracts"),
6    networks: {
7      develop: {
8        port: 8545
9      }
10   },
11   compilers: {
12     solc: {
13       version: "^0.6.0"
14     }
15   }
```

- ❑ Setelah mengatur compiler dari trufllnya
- ❑ Membuka powershell lagi dan melakukan perintah "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 :Supply 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 memasukkan script sebelumnya
- ❑ Lalu menambahkan render pada App.js
- ❑ Dan menambahkan fungsi handleChange dan handleSubmit

```
render() {
  if (!this.state.loaded) {
    return <div>Loading Web3, accounts, and contract...</div>;
  }
  return (
    <div className="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.handleChange} />
      Item Name: <input type="text" name="itemName" value={this.state.itemName} onChange={this.handleChange} />
      <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);
};

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

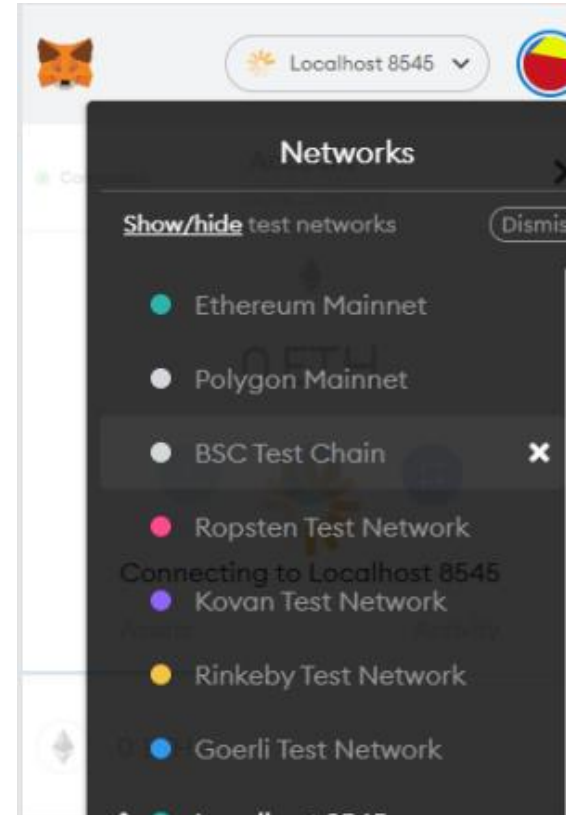
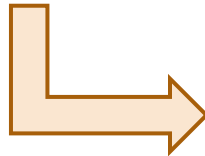
LAB :Supply Chain Project

```
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.
```

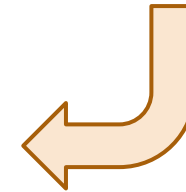
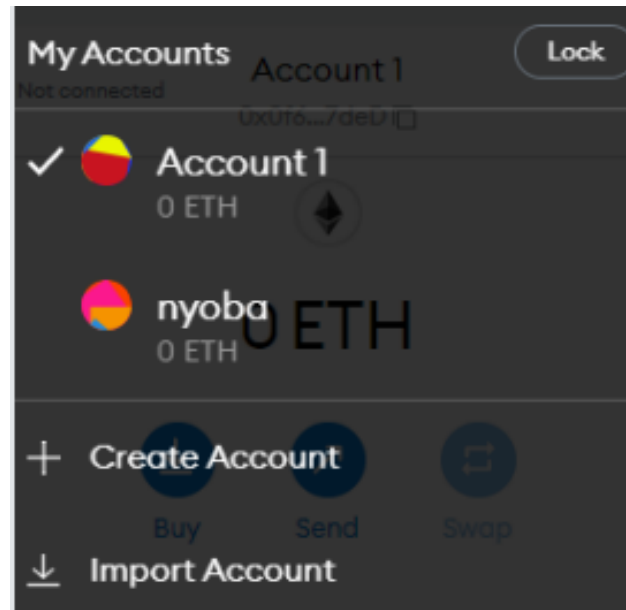
- ❑ Melakukan npm start pada powershell
- ❑ Dan beralih pada metamask untuk mengubah network menjadi “localhost 8545”



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 :Supply Chain Project

Import Account

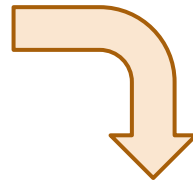
Imported accounts will not be associated with your originally created MetaMask account Secret Recovery Phrase. Learn more about imported accounts [here](#)

Select Type Private Key ▼

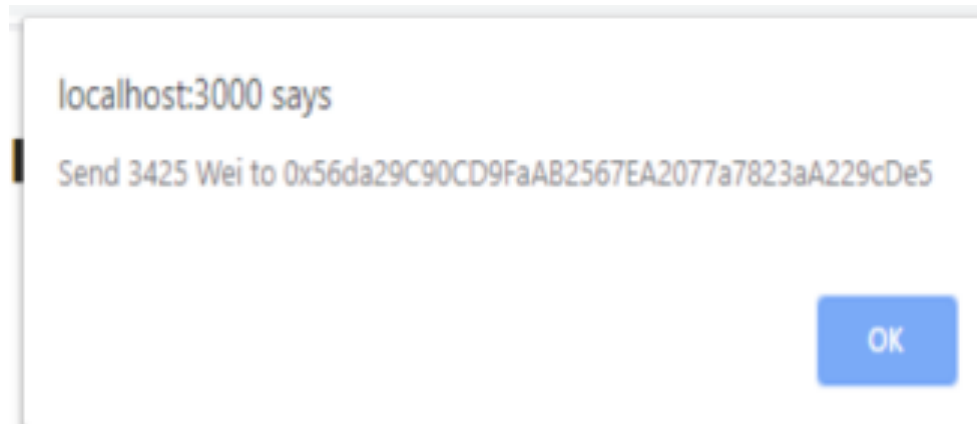
Enter your private key string here:

.....

Cancel Import



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



LAB :Supply 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

Send


✓ Account 1

0x0f6308521aded25c38d08005a449ea1a359e7ded

✕

Insufficient funds for gas

Asset:

 ETH

Balance: 0 ETH

Amount:

0 ETH

No Conversion Rate Available

↑↓

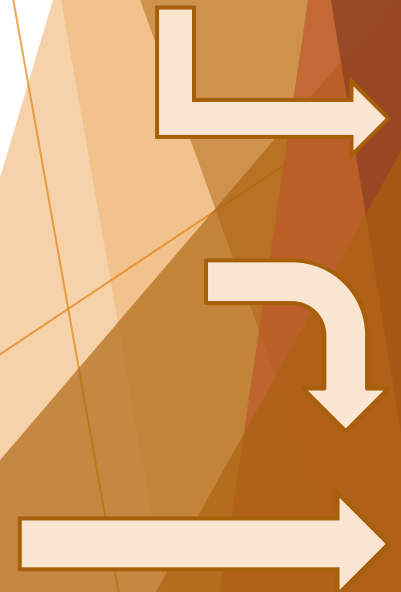
Max

Insufficient funds.

Cancel

Next

- ❑ Maka setelah itu memindahkan yang didapat dari truffle pada wallet

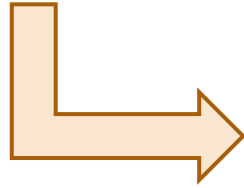


LAB :Supply Chain Project

```
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> □

