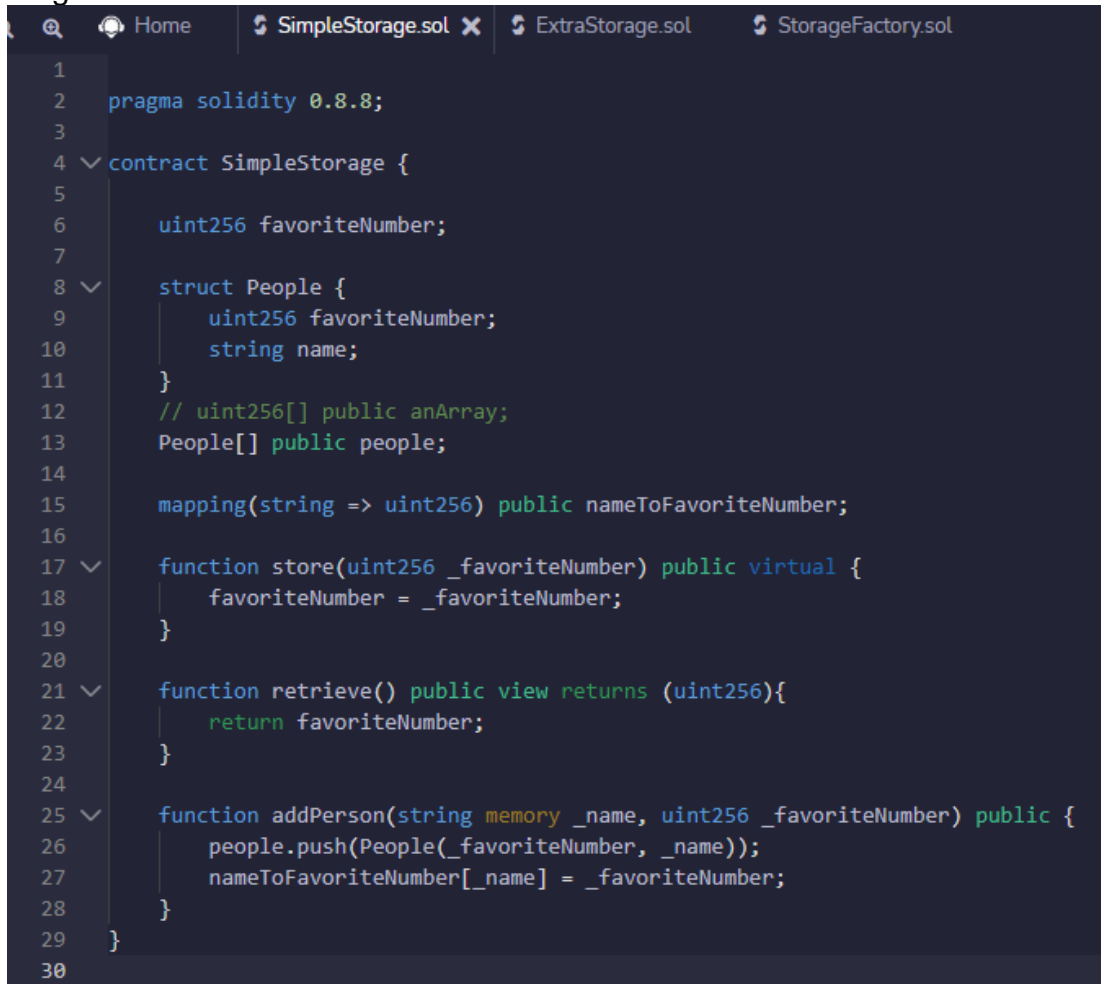


### Remix storage factory :

Interaksi antara sebuah contract dengan contract lainnya merupakan sebuah hal yang penting ketika bekerja dengan solidity dan smart contract, kemampuan ini disebut dengan composability.

Pertama kita membutuhkan sebuah file SimpleStorage.sol, dimana file ini memiliki fungsi untuk membuat sebuah smart contract untuk kita.



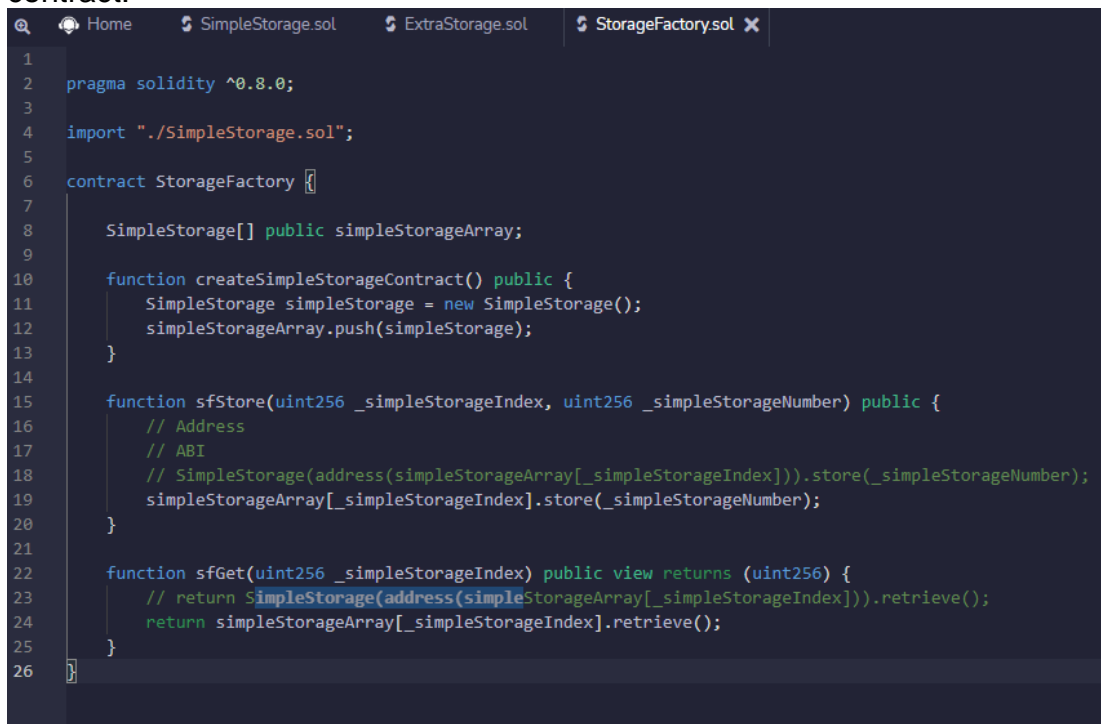
```
1
2  pragma solidity 0.8.8;
3
4  contract SimpleStorage {
5
6      uint256 favoriteNumber;
7
8      struct People {
9          uint256 favoriteNumber;
10         string name;
11     }
12     // uint256[] public anArray;
13     People[] public people;
14
15     mapping(string => uint256) public nameToFavoriteNumber;
16
17     function store(uint256 _favoriteNumber) public virtual {
18         favoriteNumber = _favoriteNumber;
19     }
20
21     function retrieve() public view returns (uint256){
22         return favoriteNumber;
23     }
24
25     function addPerson(string memory _name, uint256 _favoriteNumber) public {
26         people.push(People(_favoriteNumber, _name));
27         nameToFavoriteNumber[_name] = _favoriteNumber;
28     }
29 }
30
```

Dibuat juga ExtraStorage.sol dengan adanya inheritance dari SimpleStorage sebagai child contract dengan fungsi-fungsi dari parent contract, dan fungsi untuk menambahkan nilai store dengan 5.



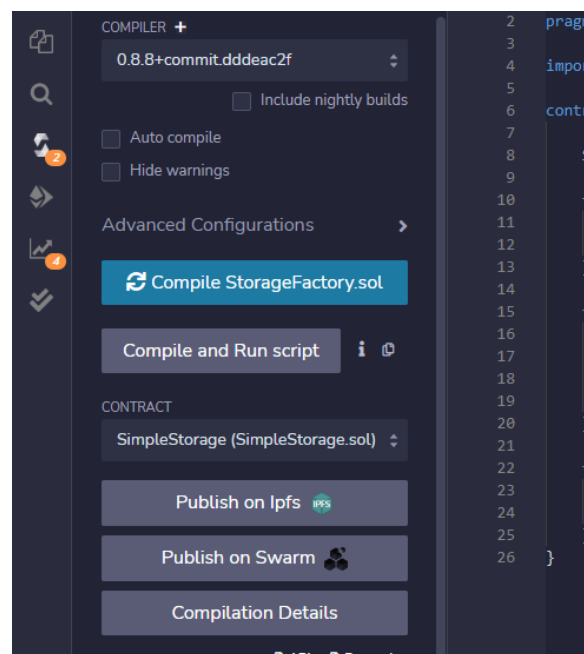
```
1
2  pragma solidity 0.8.8;
3
4  import "./SimpleStorage.sol";
5
6  contract ExtraStorage is SimpleStorage {
7      function store(uint256 _favoriteNumber) public virtual override {
8          favoriteNumber = _favoriteNumber + 5;
9      }
10 }
11
```

Pada StorageFactory.sol terdapat fungsi untuk mendeploy dan berinteraksi dengan kontrak lainnya. Fungsi sfStore() akan mengatur simple storage yang ada beserta datanya dalam sebuah contract, dan sfGet() digunakan untuk menampilkan isi smart contract.



```
1  pragma solidity ^0.8.0;
2
3
4  import "./SimpleStorage.sol";
5
6  contract StorageFactory {
7
8      SimpleStorage[] public simpleStorageArray;
9
10     function createSimpleStorageContract() public {
11         SimpleStorage simpleStorage = new SimpleStorage();
12         simpleStorageArray.push(simpleStorage);
13     }
14
15     function sfStore(uint256 _simpleStorageIndex, uint256 _simpleStorageNumber) public {
16         // Address
17         // ABI
18         // SimpleStorage(address(simpleStorageArray[_simpleStorageIndex])).store(_simpleStorageNumber);
19         simpleStorageArray[_simpleStorageIndex].store(_simpleStorageNumber);
20     }
21
22     function sfGet(uint256 _simpleStorageIndex) public view returns (uint256) {
23         // return SimpleStorage(address(simpleStorageArray[_simpleStorageIndex])).retrieve();
24         return simpleStorageArray[_simpleStorageIndex].retrieve();
25     }
26 }
```

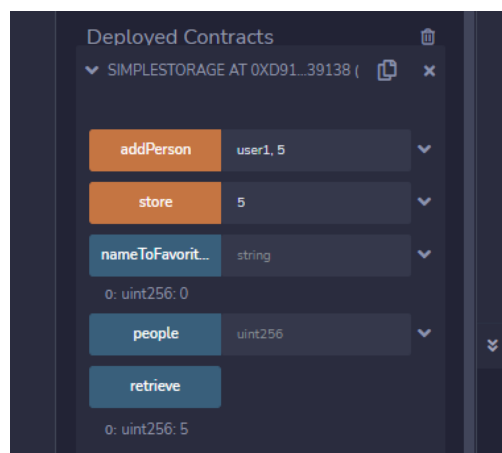
Setelah semuanya siap, kita akan mencoba membuat sebuah smart contract. Perama kita perlu mencompile program terlebih dahulu



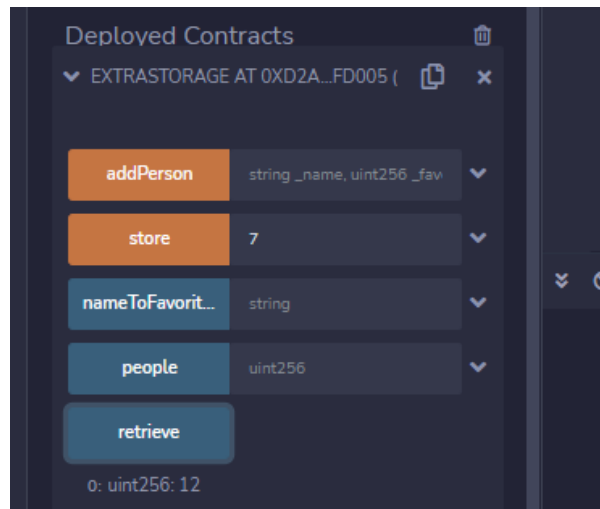
Informasi dari berbagai contract yang sudah dcompile seperti interaksi apa saja yang tersedia dapat kita lihat dalam menu compilation details



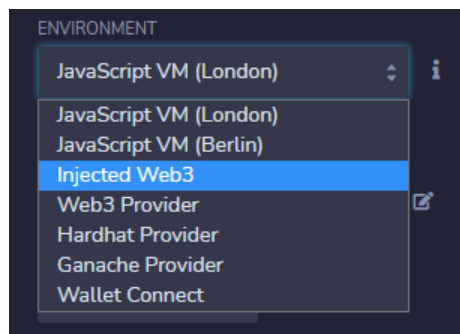
Untuk membuat contract baru, pada fungsi addPerson perlu kita isi dengan nama pengguna, serta sebuah angka. Setelah itu isi fungsi store dengan sebuah nilai juga. Maka angka tersebut akan tersimpan dalam smart contract, dan dapat di check dengan fungsi retrieve.



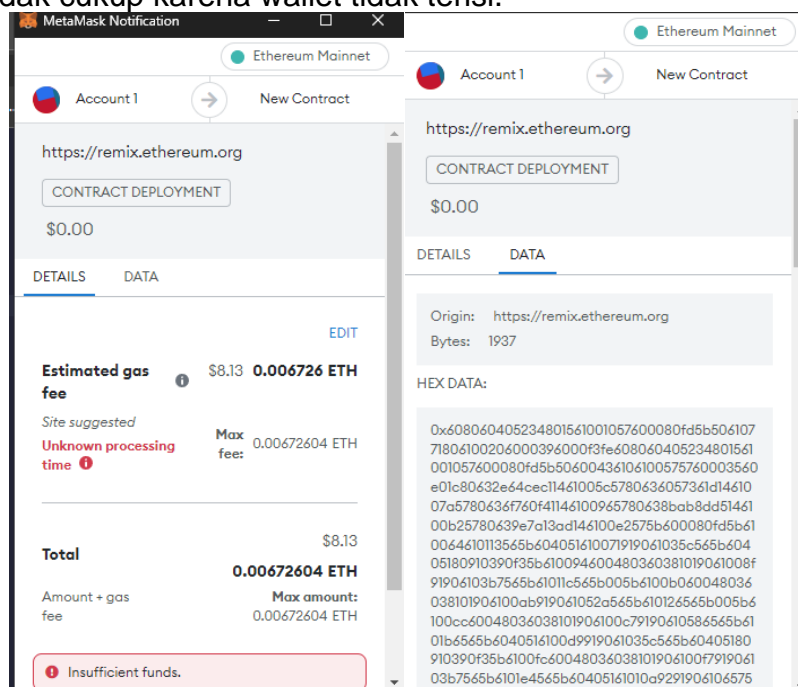
ExtraStorage juga memiliki interaksi yang sama dengan SimpleStorage, namun memiliki fungsi tambahan untuk menambahkan nilai store dengan 5. Caranya dengan mendeploy contract terlebih dahulu, dan mengisi nilai store dengan angka. Dapat dilihat nilai 7 setelah di store akan menjadi 12.



Untuk mencoba penggunaan smart contract secara langsung, dapat menggunakan metamask dengan wallet yang telah dibuat dengan mengubah environment menjadi Injected Web3.



Lalu akan ada notifikasi masuk kedalam metamask mengenai contract deployment dan datanya, yang membutuhkan biaya gas. Dapat dilihat untuk kali ini biaya deployment tidak cukup karena wallet tidak terisi.



Untuk menambah ethereum pada wallet, dapat menggunakan network seperti Rinkeby dan menambah isi ethereum wallet

**i** Your wallet is connected to Ethereum Rinkeby, so you are requesting Ethereum Rinkeby LII

Wallet address


0x93471fe3e9c0f59ddfb143a5dbed5112e4e4723e

Request type

☐ 20 test LINK

☒ 0.1 test ETH


☒ I am human

 hCaptcha  
Privacy - Terms

**Send request**

**?** Need more testnet ETH? Get ETH from [Ethereum Rinkeby Faucet](#)

Setelah itu wallet akan terisi di network rinkeby



Ethereum Rinkeby

Connected

Account 1

0x934...723e

0.1 ETH

Buy

Send

Swap

Assets

Activity

Receive

Jul 8 · From: 0xa7a...24a8

0.1 ETH

0.1 ETH

Need help? Contact [MetaMask Support](#)

Ethereum Rinkeby

Account 1

New Contract

0x934...723e

Copy address to clipboard

CONTRACT DEPLOYMENT

DETAILS

DATA

EDIT

Estimated gas fee

0.00115921

0.001159 ETH

Site suggested

Very likely in < 15 seconds

Max fee: 0.00115921 ETH

Total

0.00115921

0.00115921 ETH

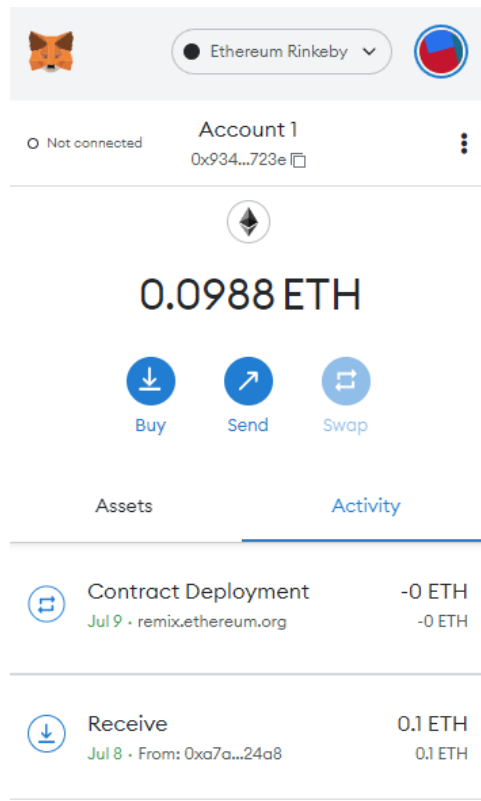
Amount + gas fee

Max amount: 0.00115921 ETH

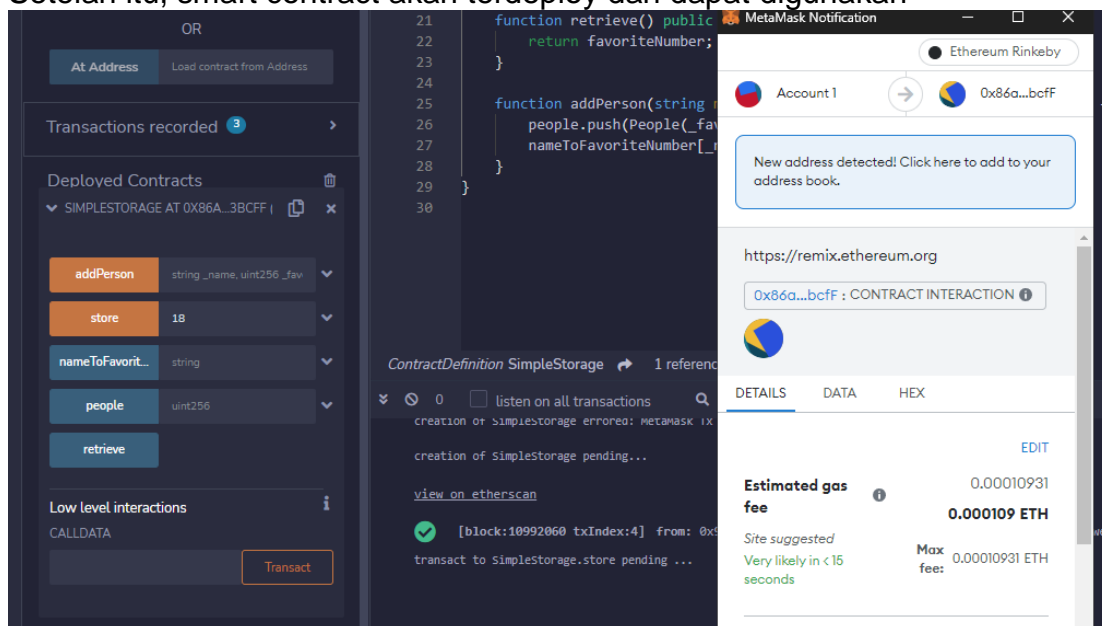
Reject

Confirm

Dan smart contract dapat di deploy dengan gas yang cukup



Setelah itu, smart contract akan terdeploy dan dapat digunakan



Transaksi dapat dilihat dengan etherscan pada pilihan view on etherscan

Transaction Details

Overview

State

[ This is a Rinkeby **Testnet** transaction only ]

Transaction Hash:	0xe23758b36fff82dcf21fb0b6c4cc375451da066d2ce12a92a5f57b4ea0de20c9
Status:	Success
Block:	10992078 2 Block Confirmations
Timestamp:	22 secs ago (Jul-09-2022 03:54:55 AM +UTC)
From:	0x93471fe3e9c0f59ddfb143a5dbed5112e4e4723e
To:	Contract 0x86a29ac501718c455fe955df8b987081a483bcff
Value:	0 Ether (\$0.00)
Transaction Fee:	0.000109310000349792 Ether (\$0.00)
Gas Price:	0.000000002500000008 Ether (2.500000008 Gwei)

Click to see More