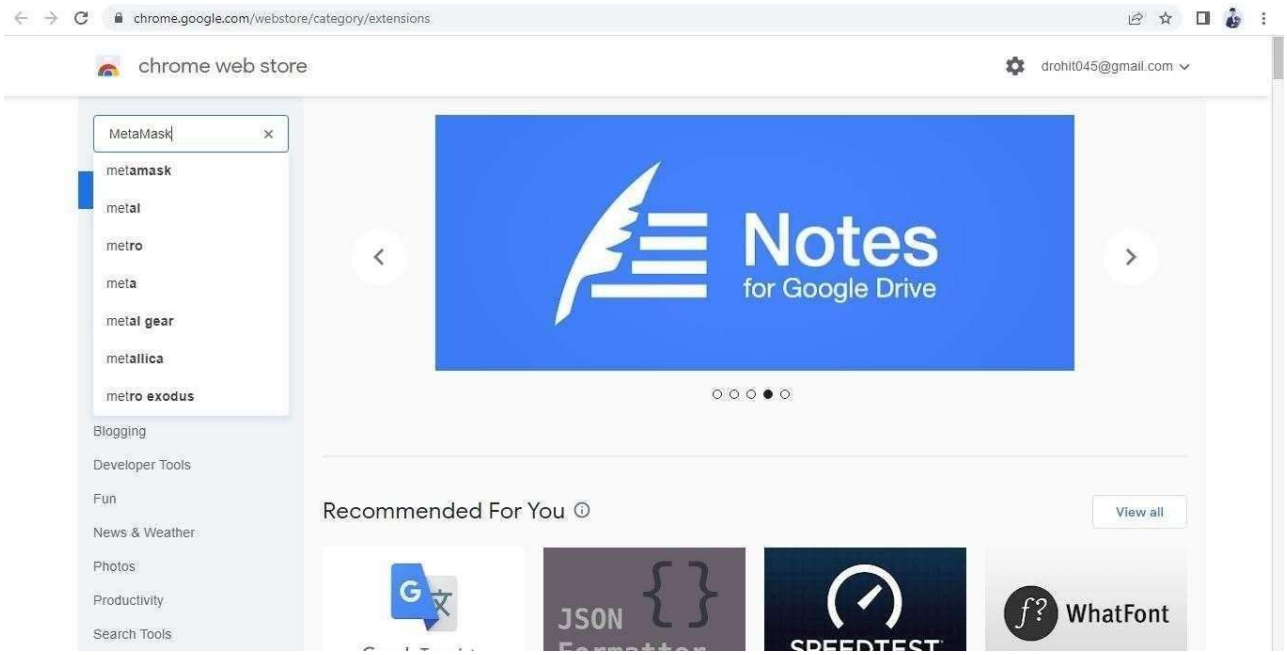
Lab Assignment No.01

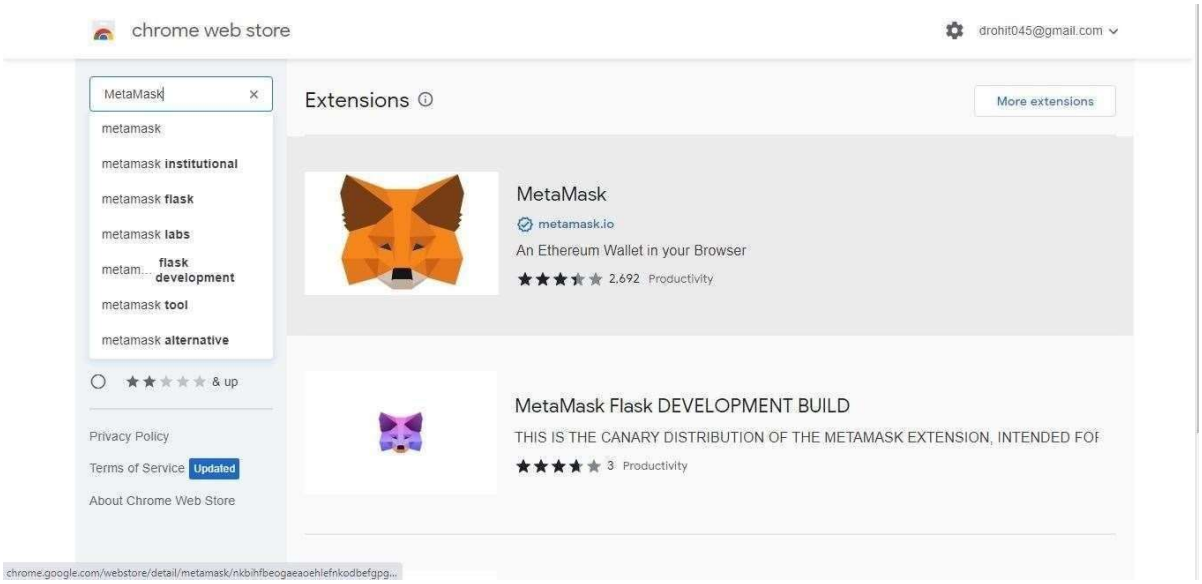
Installation of MetaMask:

Step 1: Go to Chrome Web Store Extensions Section.

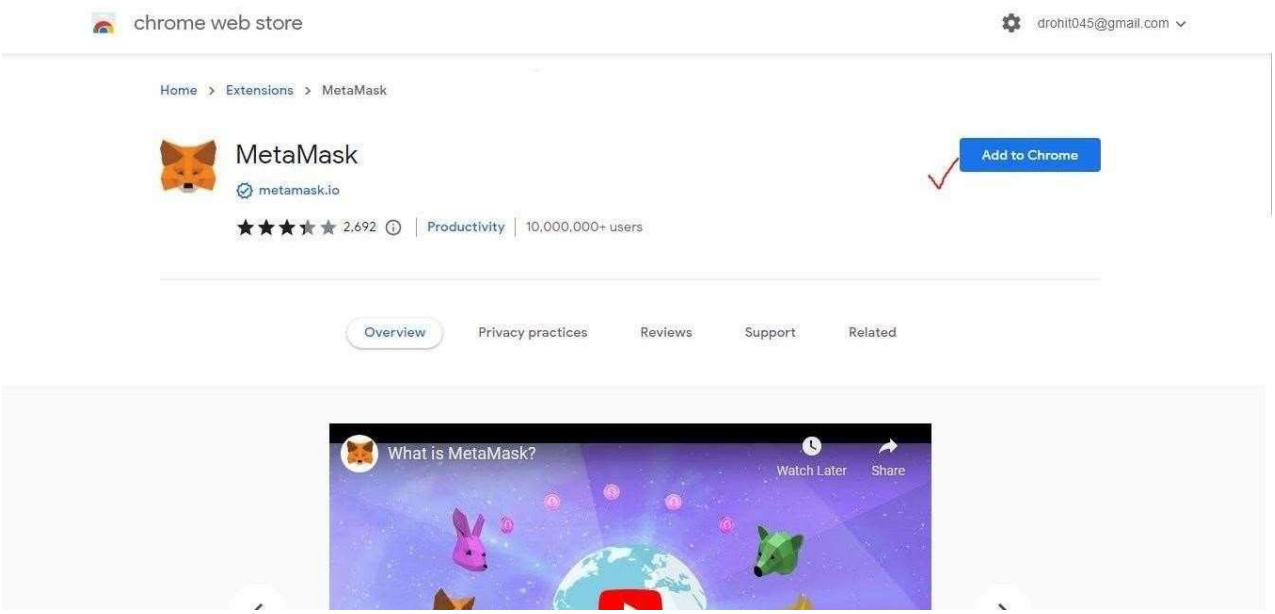
Step 2: Search MetaMask.



Step 3: Check the number of downloads to make sure that the legitimate MetaMask is being installed

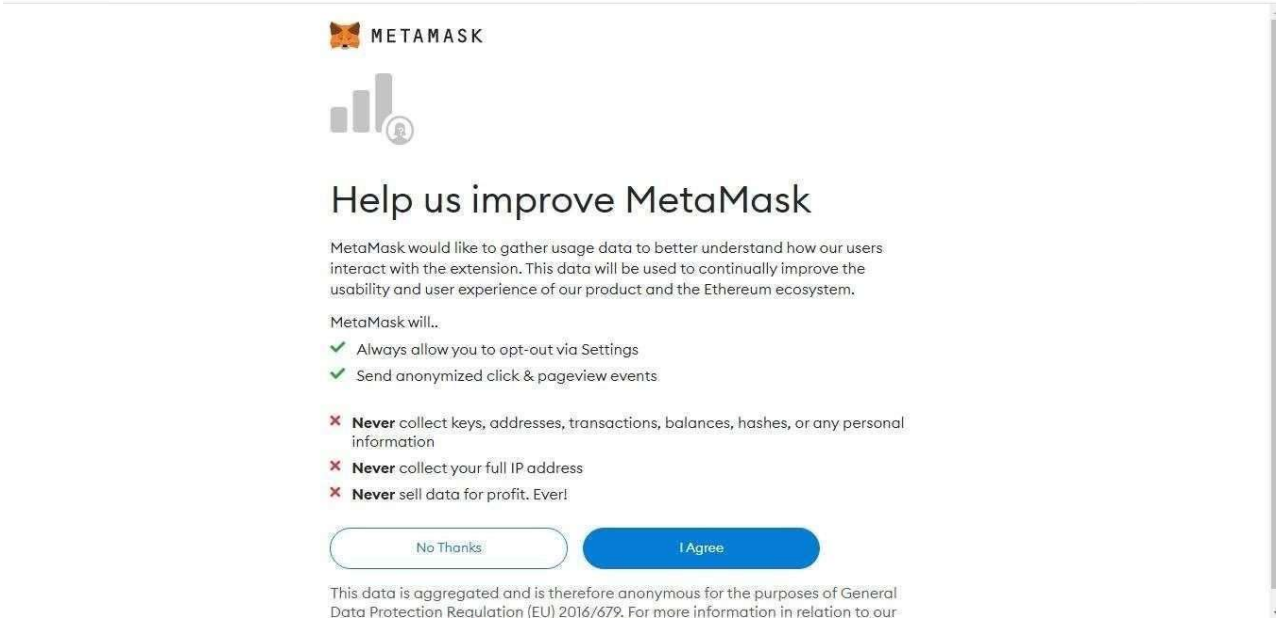


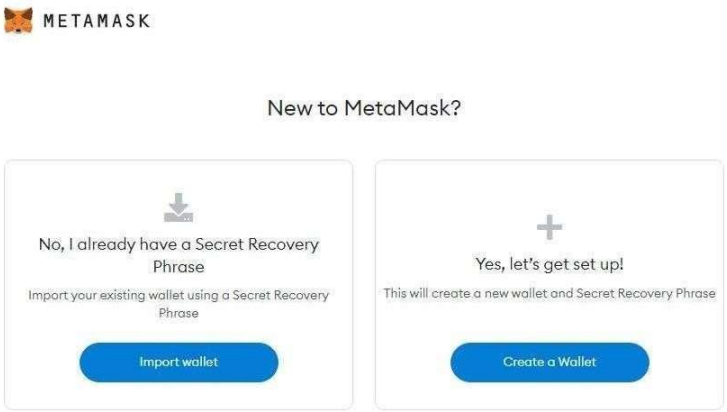
Step 4: Click the Add to Chrome button.



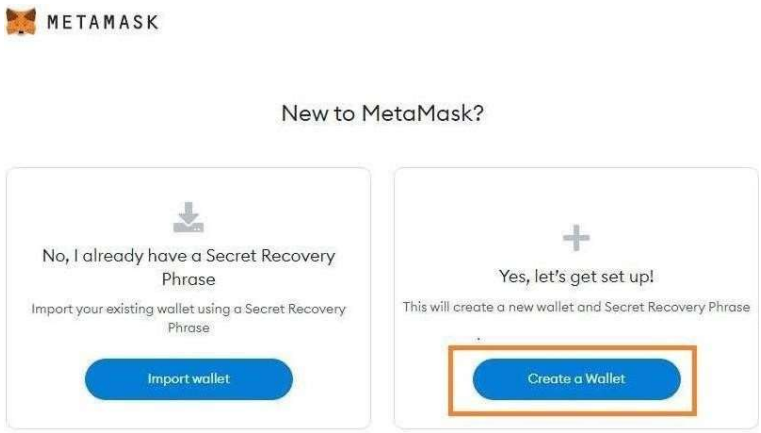
Step 5: Once installation is complete this page will be displayed. Click on the Get Started button.

Step 6: This is the first time creating a wallet, so click the Create a Wallet button. If there is alreadya wallet then import the already created using the Import Wallet button.



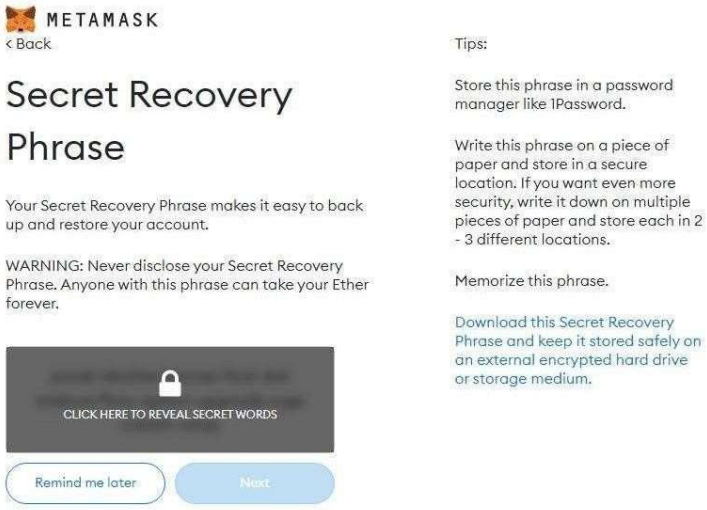


Step 7: Click I Agree button to allow data to be collected to help improve MetaMask or else click the No Thanks button. The wallet can still be created even if the user will click on the No Thanks button.

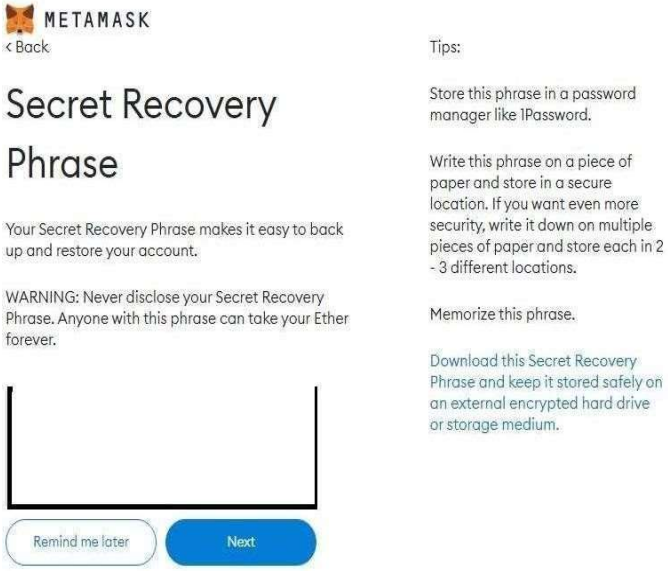


Step 8: Create a password for your wallet. This password is to be entered every time the browser is launched and wants to use MetaMask.

Step 9: Click on the dark area which says *Click here to reveal secret words* to get your secret phrase.

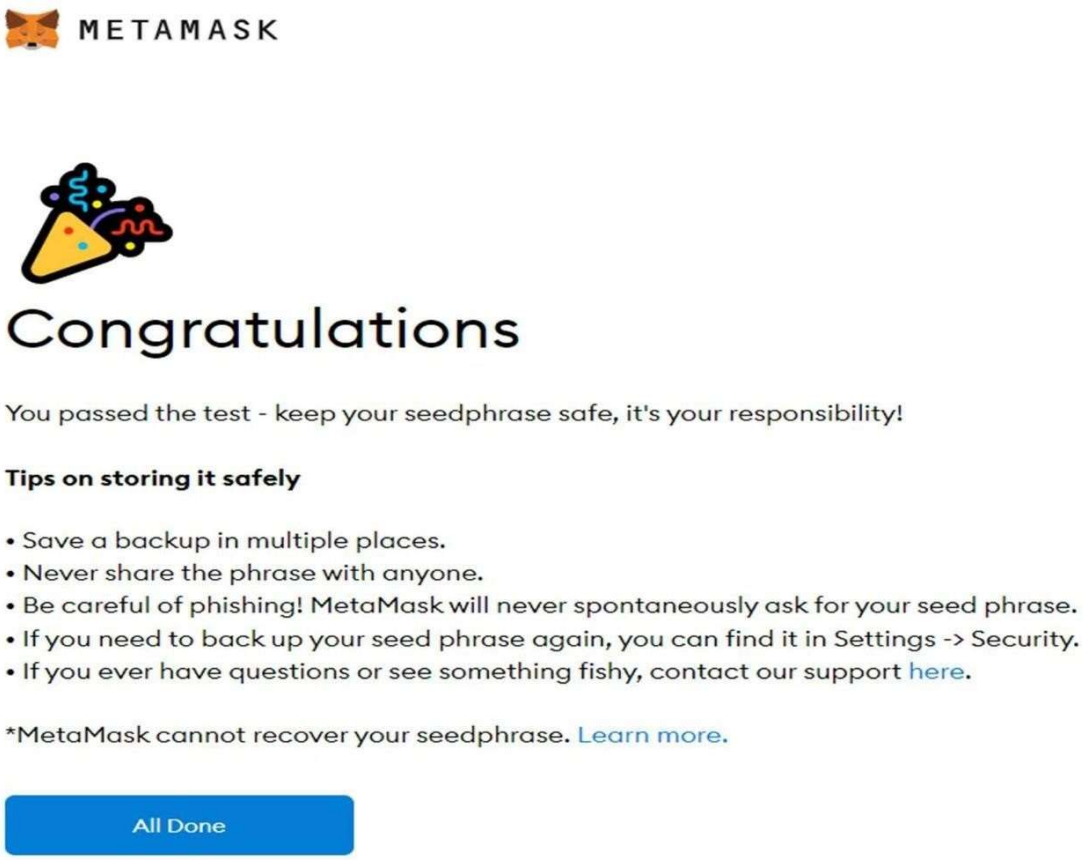


Step 10: This is the most important step. Back up your secret phrase properly. Do not store your secret phrase on your computer. The secret phrase is the only way to access your wallet if you forget your password. Once done click the *Next* button.

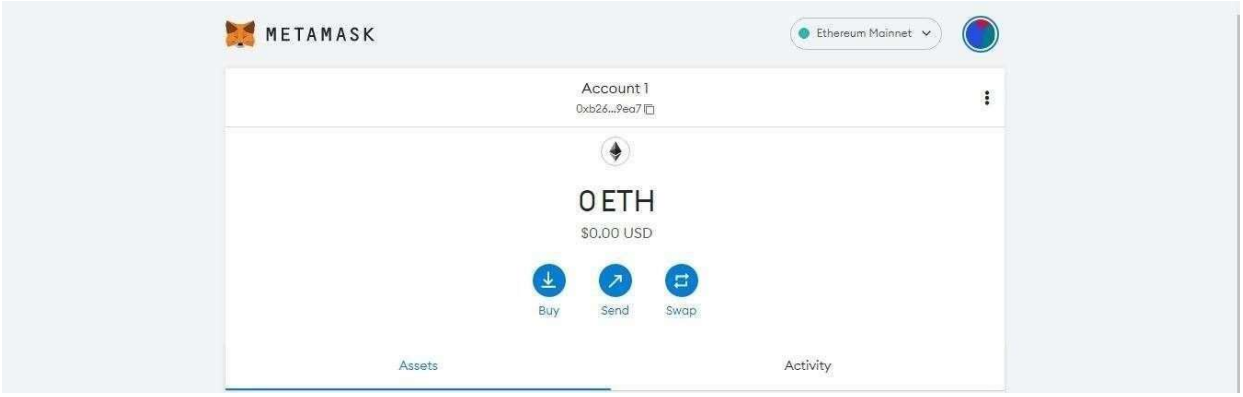


Step 11: Click the buttons respective to the order of the words in your seed phrase. In other words, type the seed phrase using the button on the screen. If done correctly the *Confirm* button should turn blue.

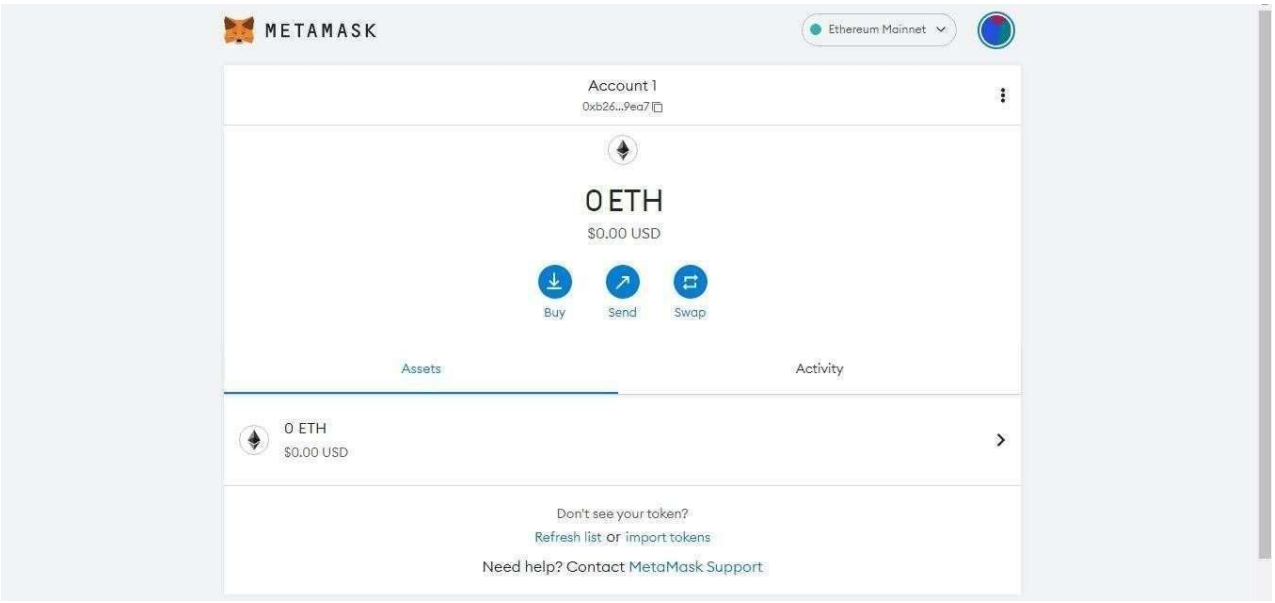
Step 12: Click the *Confirm* button.



Step 13: One can see the balance and copy the address of the account by clicking on the *Account1* area.



Step 14: One can access MetaMask in the browser by clicking the Foxface icon on the top right.If the Fox face icon is not visible, then click on the puzzle piece icon right next to it

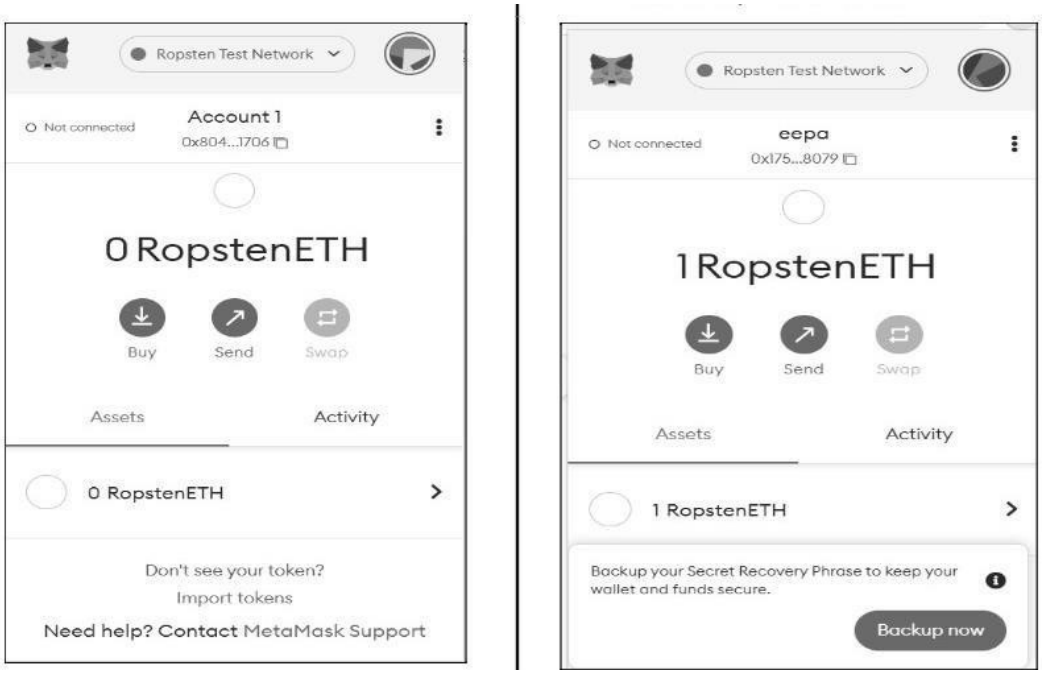


Lab Assignment No.02

Steps for transaction in Metamask:

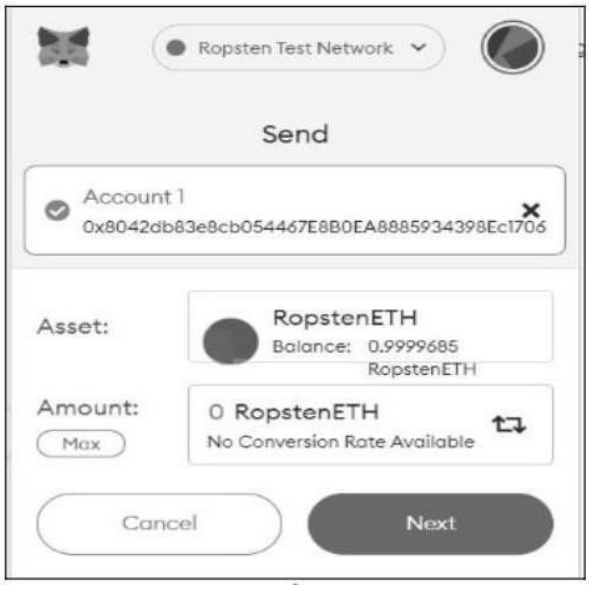
Step 1: Login to the MetaMask Account and checked the acoount Before transaction, Account 01 is Having 0 RopstenETH.

Step 2: Login to the MetaMask Account and checked the acoount Before transaction, “eppa” Account 02 is having 1 RopstenETH. Start transaction from the “eepa”. Click on send.



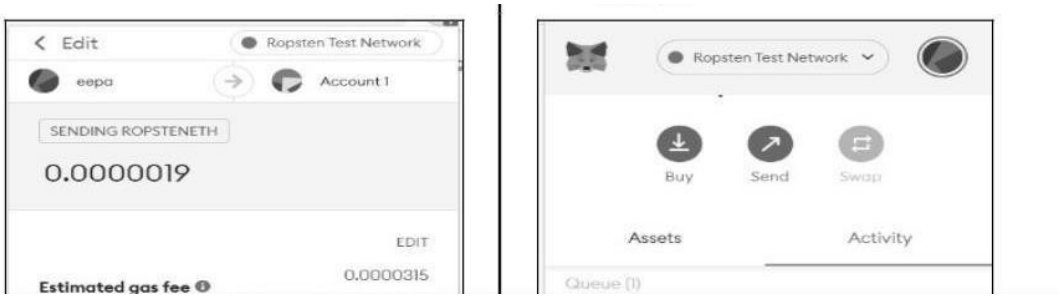
Step 3: Enter the public address of “Account 1”

Step 4: Click the Balance Amount in Assets and Enter the Amount to send the ETH. Check the Details of the Asset and Amount. Click on Next Button.

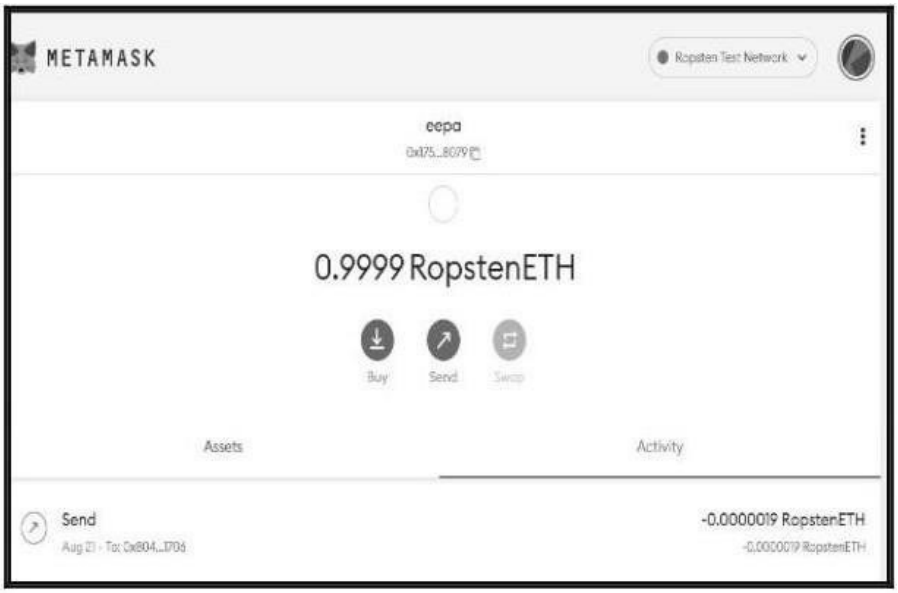


Step 5: Click on confirm Button.

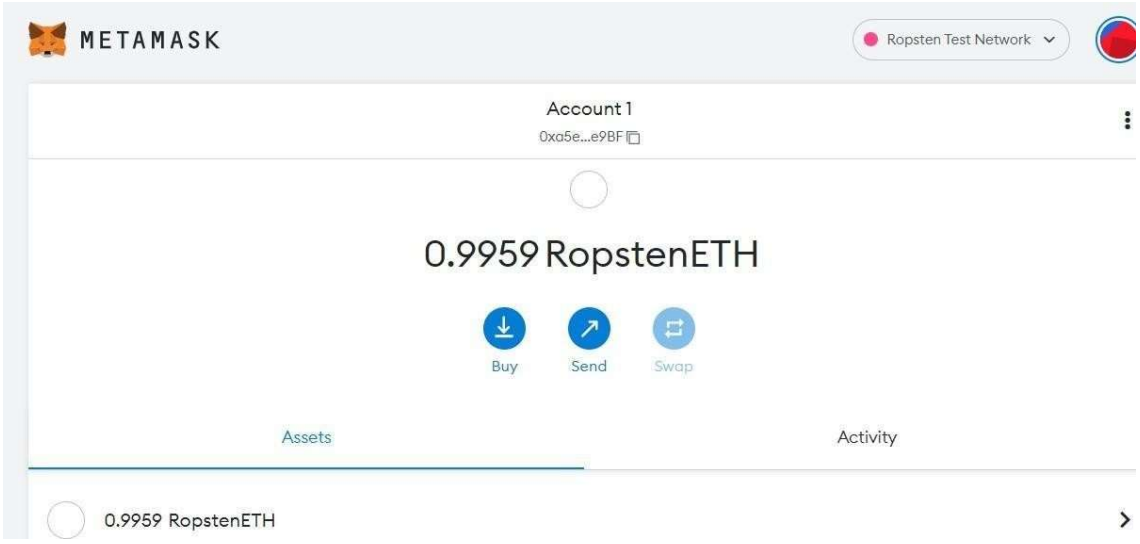
Step 6: Transaction status will show 'Pending' for few times wait.

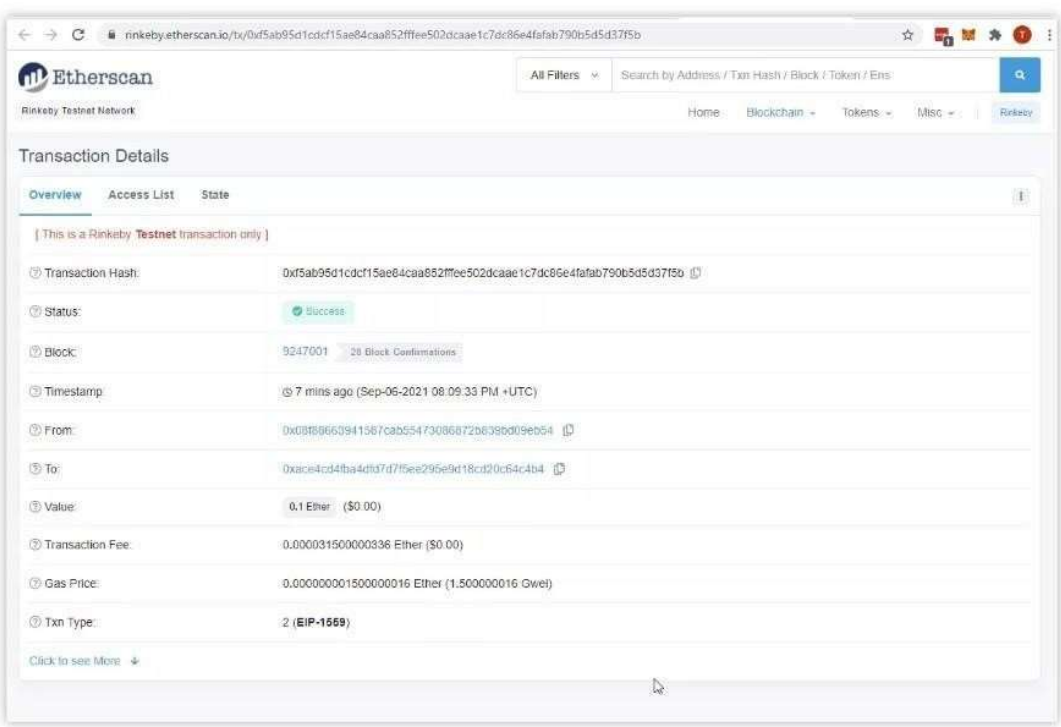


Step 7: Transaction is successfully done. Account 1 Received ETH



Step 8: Track the transaction using ETH.

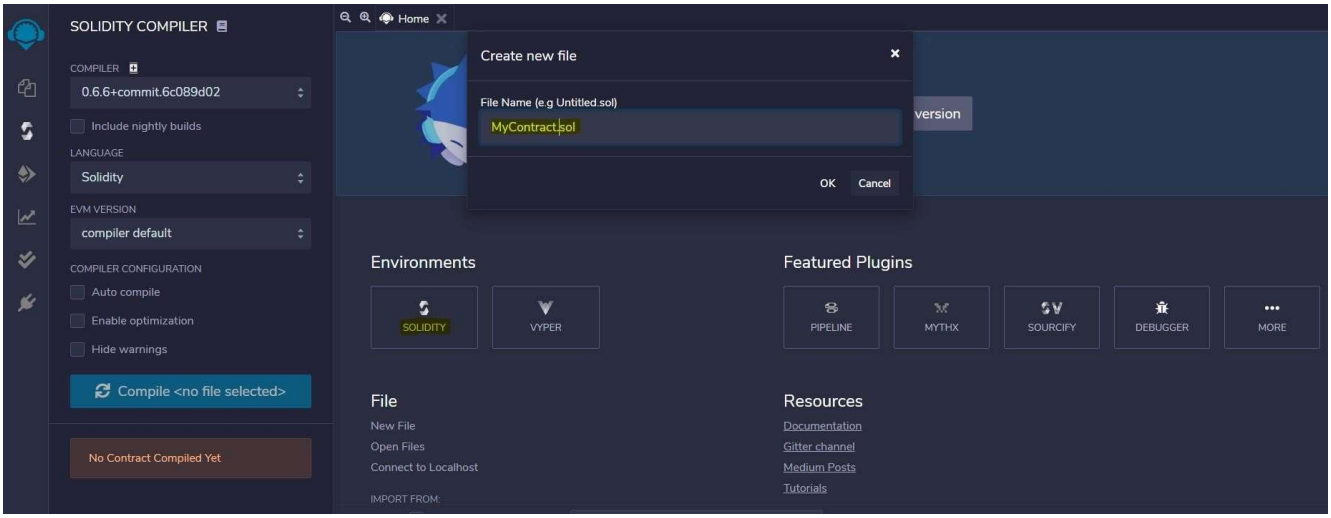




Lab Assignment No.03

Step 1: Open Remix-IDE.

Step 2: Select File Explorer from the left side icons and select Solidity in the environment. Click on New option below the Solidity environment. Enter the file name as MyContract.sol and Click on the OK button.

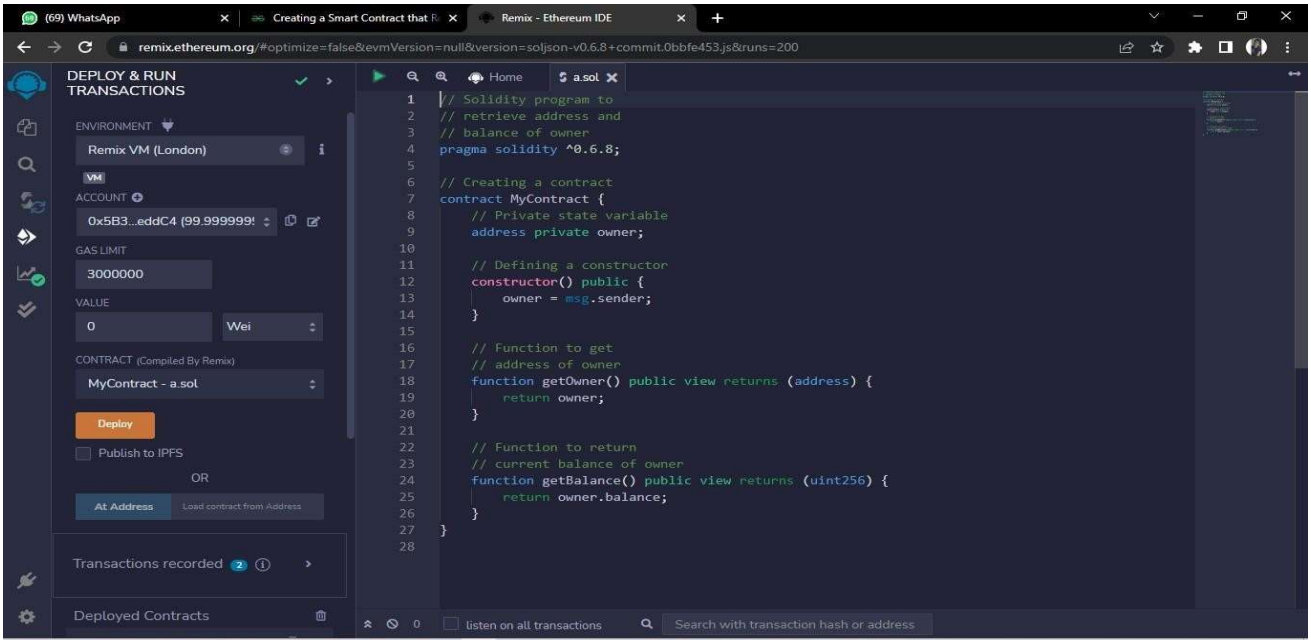


Step 3: Enter the following Solidity Code.

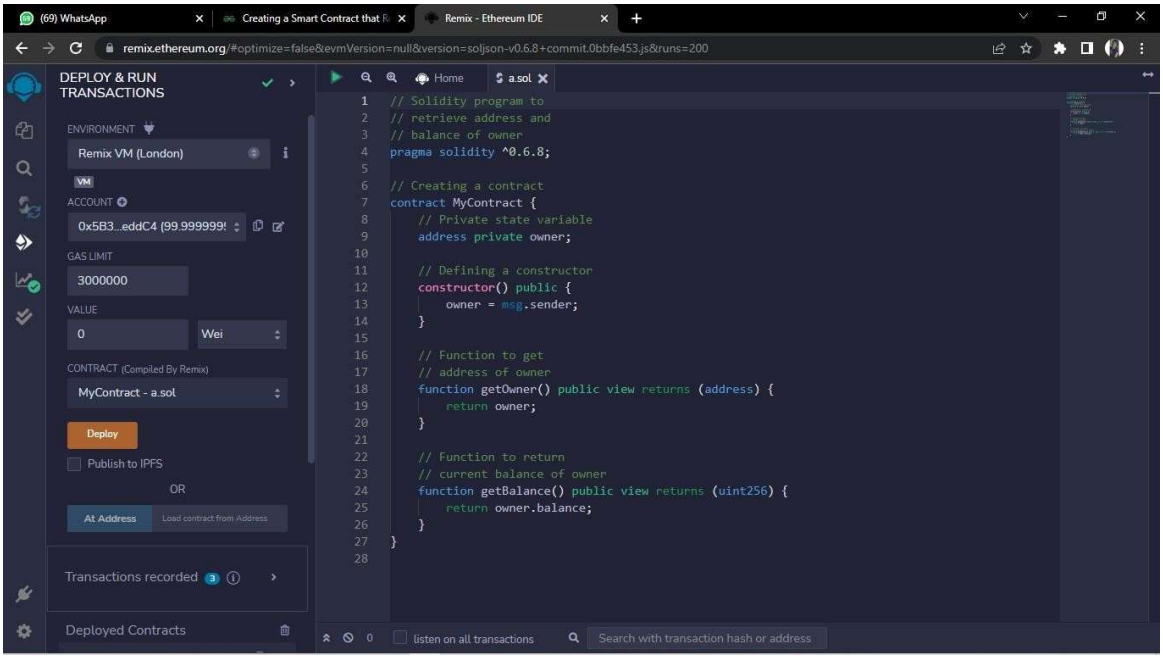
```
// Solidity program to
// retrieve address and
// balance of owner
pragma solidity ^0.6.8;

// Creating a contract
contract MyContract
{
        // Private state variable
        address private owner;

        // Defining a constructor
        constructor() public{
                owner=msg.sender;
        }

        // Function to get
        // address of owner
        function getOwner(
        ) public view returns (address) {
                return owner;
        }
```

```
        // Function to return
        // current balance of owner
        function getBalance(
        ) public view returns(uint256){
                return owner.balance;
        }
}
```

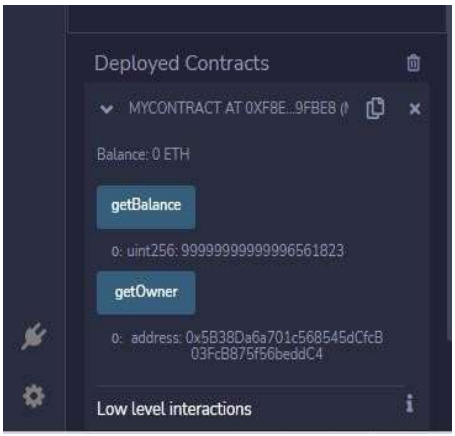
Step 4: Compile the file MyContract.sol from the Solidity Compiler tab.



Step 5: Deploy the smart contract from the Deploy and Run Transaction tab and you will get the balance and address of the owner.

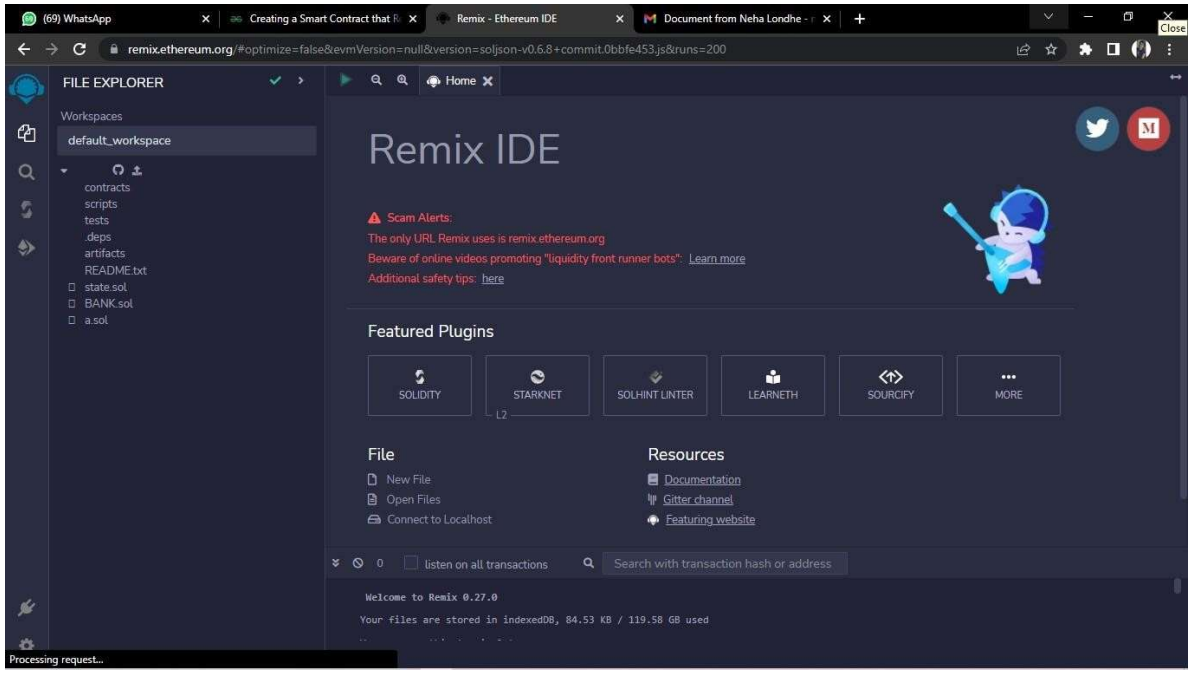


Step 6: The output below shows the address and the balance of the owner.



Lab Assignment No.04

Step 1: Create smart contract using solidity programming.

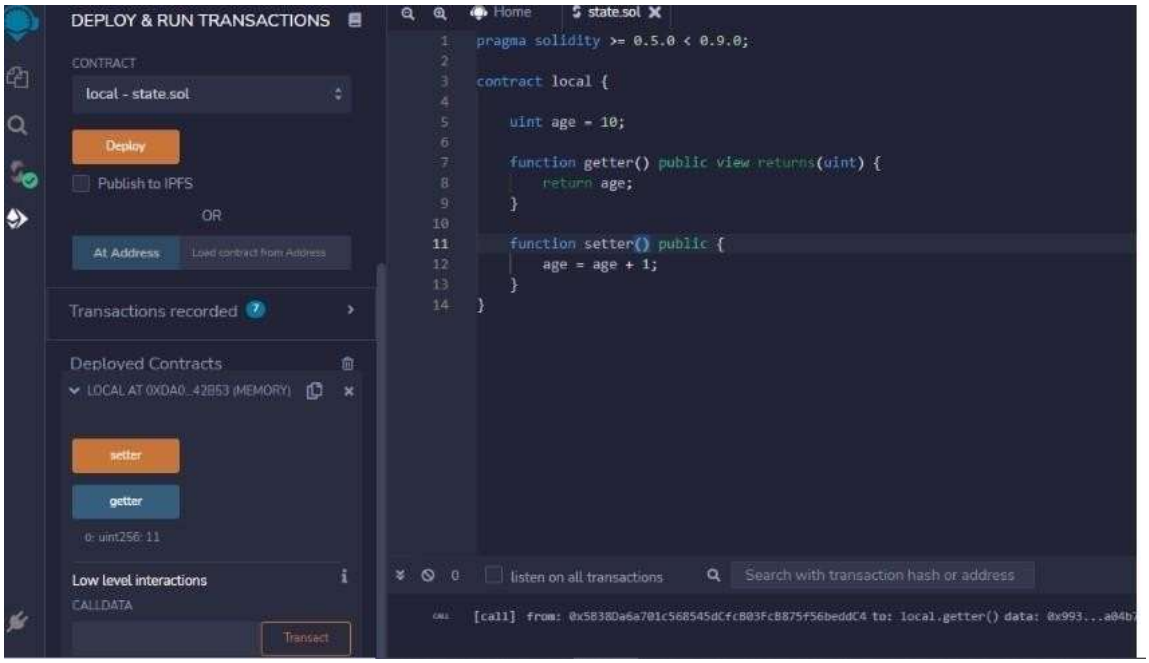


Functions in Solidity

1. **Simple public function.**

```
pragma solidity >= 0.5.0 < 0.9.0;
contract local {
uint age = 10;
function getter() public view returns(uint) {
return age;
}
function setter() public {
age = age + 1;
}
}
```

Step 2: After writing the above code on Remix IDE in a new file with sol extension, you can compile the code, visit the deploy section, and deploy the code to observe the deploy section output as shown below. The value will increase asyou click the setter and getter function buttons.



2. Constructor in solidity

```
pragma solidity >= 0.5.0 < 0.9.0;
contract local {
    uint public count;
    constructor(uint new_count) {
        count = new_count;
    }
}
```

3. Loops:

```
pragma solidity >= 0.5.0 < 0.9.0;
contract Loops
{
uint [3] public arr; uint public count;
        function Whileloop() public
        {
                while(count < arr.length)
                {
                arr[count] = count; count++;
                }
        }
function Forloop() public {
        for(uint i=count; i<arr.length; i++)
        {
                arr[count] = count;
                count++;
        }
}
```

If-else Statements in Solidity:

```
pragma solidity >= 0.5.0 < 0.9.0;
contract Array {
	function check(int a) public pure returns(string memory) {
	string memory value;
	if(a > 0) {
		value = "Greater Than zero";
	}
	else if(a == 0) {
	value = "Equal to zero";
	}
	else {
	value = "Less than zero";
}
return value;
}
}
```

4. **Arrays:**

```
pragma solidity >= 0.5.0 < 0.9.0;
contract Array {
    uint [4] public arr = [10, 20, 30, 40];
    function setter(uint index, uint value) public {
        arr[index] = value;
    }
    function length() public view returns(uint) {
        return arr.length;
    }
}
```

5. **For dynamic array:**

```
pragma solidity >= 0.5.0 < 0.9.0;
contract Array {
    uint [] public arr;
    function PushElement(uint item) public {
        arr.push(item);
    }
    function Length() public view returns(uint) {
        return arr.length;
    }
    function PopElement() public {
        arr.pop();
    }
}
```

Structure in Solidity:

```
pragma solidity >= 0.5.0 < 0.9.0;
struct Student {
    uint rollNo;
    string name;
}
contract Demo {
    Student public s1;
    constructor(uint _rollNo, string memory _name) {
        s1.rollNo = _rollNo;
        s1.name = _name;
    }
    // to change the value we have to implement a setter function
    function changeValue(uint _rollNo, string memory _name) public {
        Student memory new_student = Student( {
            rollNo : _rollNo,
            name : _name
        });
        s1 = new_student;
    }
}
```

Create a Smart Contract with CRUD Functionality

```
pragma solidity ^0.5.0;
contract Crud {
        struct User {
                uint id;
                string name;
        }
        User[] public users;
        uint public nextId = 0;
        function Create(string memory name) public {
                users.push(User(nextId, name));
                nextId++;
        }
        function Read(uint id) view public returns(uint, string memory) {
                for(uint i=0; i<users.length; i++) {
                        if(users[i].id == id) {
                                return(users[i].id, users[i].name);
                        }
        }
        function Update(uint id, string memory name) public {
                 for(uint i=0; i<users.length; i++) {
                        if(users[i].id == id) {
                                users[i].name =name;
                        }
                }
        }
        function Delete(uint id) public {
                delete users[id];
        }
        function find(uint id) view internal returns(uint) {
                 for(uint i=0; i< users.length; i++) {
```