Assignment 1

Aim: Installation of Metamask and study spending Ether per transaction.

Theory:

Metamask: MetaMask is a type of Ethereum wallet that bridges the gap between the user interfaces for Ethereum (e.g. Mist browsers, DApps) and the regular web (e.g. Chrome, Firefox, websites).

Its function is to inject a JavaScript library called web3.js into the namespace of each page your browser loads. Web3.js is written by the Ethereum core team.

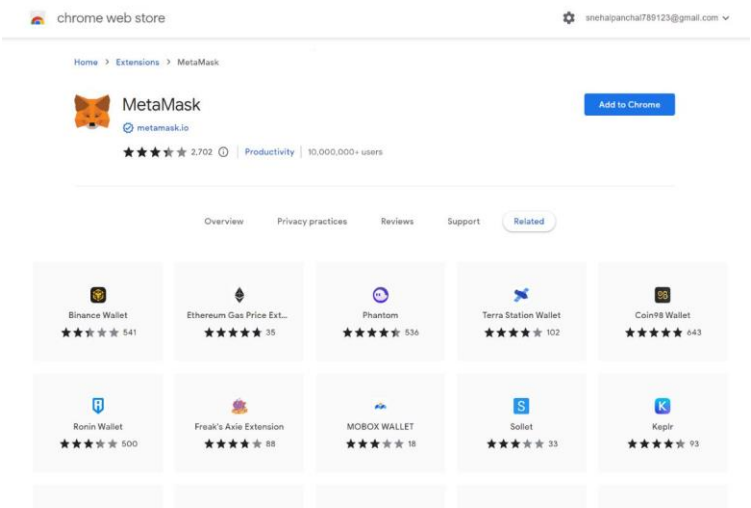
fter adding MetaMask as an extension in chrome and creating an account, set up your account as follows –

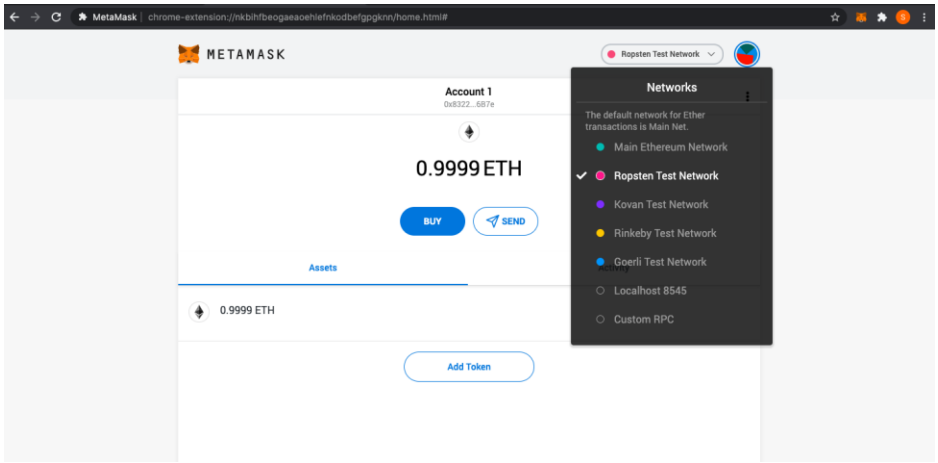
How to Install and Use Metamask

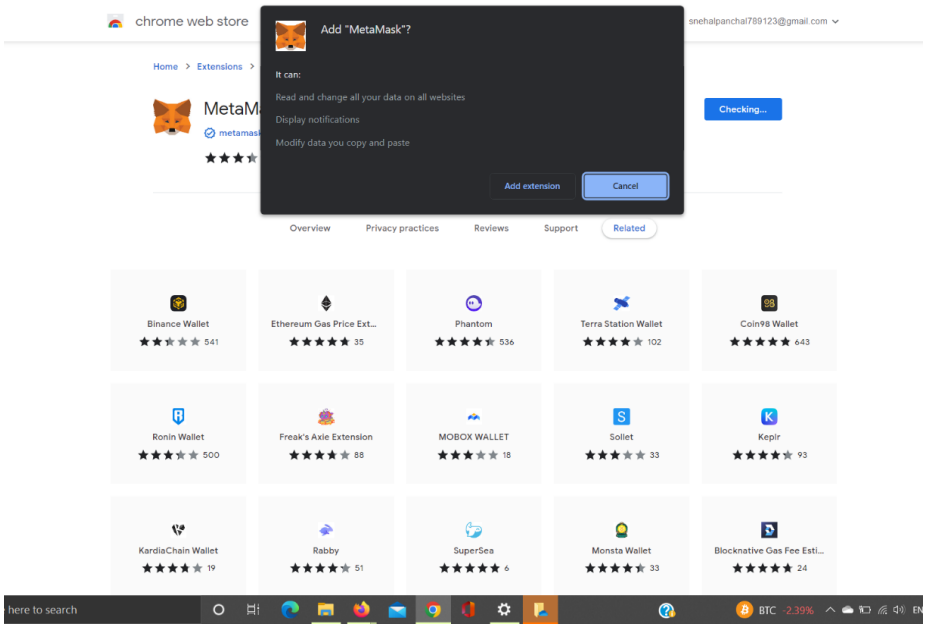
1. Go to the Metamask website.
2. Click “Get Chrome Extension” to install Metamask.
3. Click “Add to Chrome” in the upper right.
4. Click “Add Extension” to complete the installation. ...

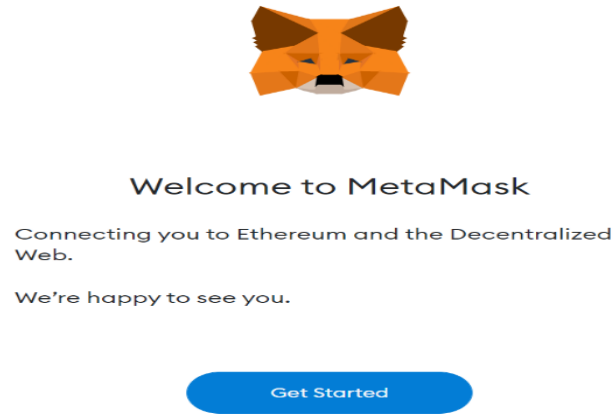
Click on the Metamask logo in the upper right hand corner of your Google chrome browse

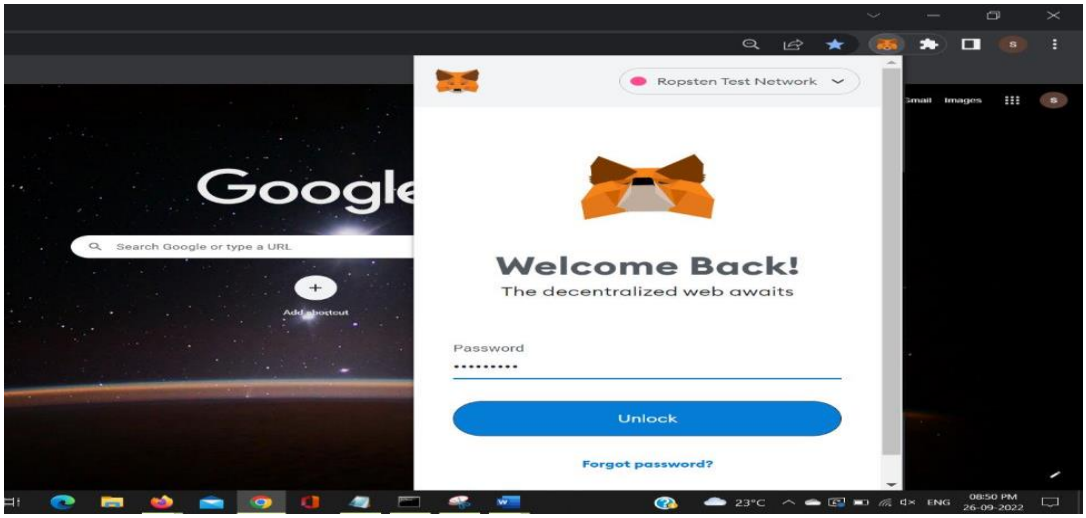
Step 1: Select *Ropsten Test Network* from a list of available networks as below:




MetaMask
chrome-extension://nkbihfbeogaeaoehlefnkodbefgpgknn/home.html#
METAMASK
Ropsten Test Network
Account 1
0x8222...687e
0.9999 ETH
BUY
SEND
Assets
0.9999 ETH
Add Token
Networks
The default network for Ether transactions is Main Net.
Main Ethereum Network
Ropsten Test Network
Kovan Test Network
Rinkeby Test Network
Goerli Test Network
Localhost 8545
Custom RPC


chrome web store
Add "MetaMask"?
It can:
Read and change all your data on all websites
Display notifications
Modify data you copy and paste
Add extension
Cancel
Checking...
Overview
Privacy practices
Reviews
Support
Related
Binance Wallet
Ethereum Gas Price Ext...
Phantom
Terra Station Wallet
Coin98 Wallet
Ronin Wallet
Freak's Axie Extension
MOBOX WALLET
Sollet
Kepir
KardiaChain Wallet
Rabby
SuperSea
Monsta Wallet
Blocknative Gas Fee Esti...
here to search
BTC -2.39%





Step 2: Request test ether form here.

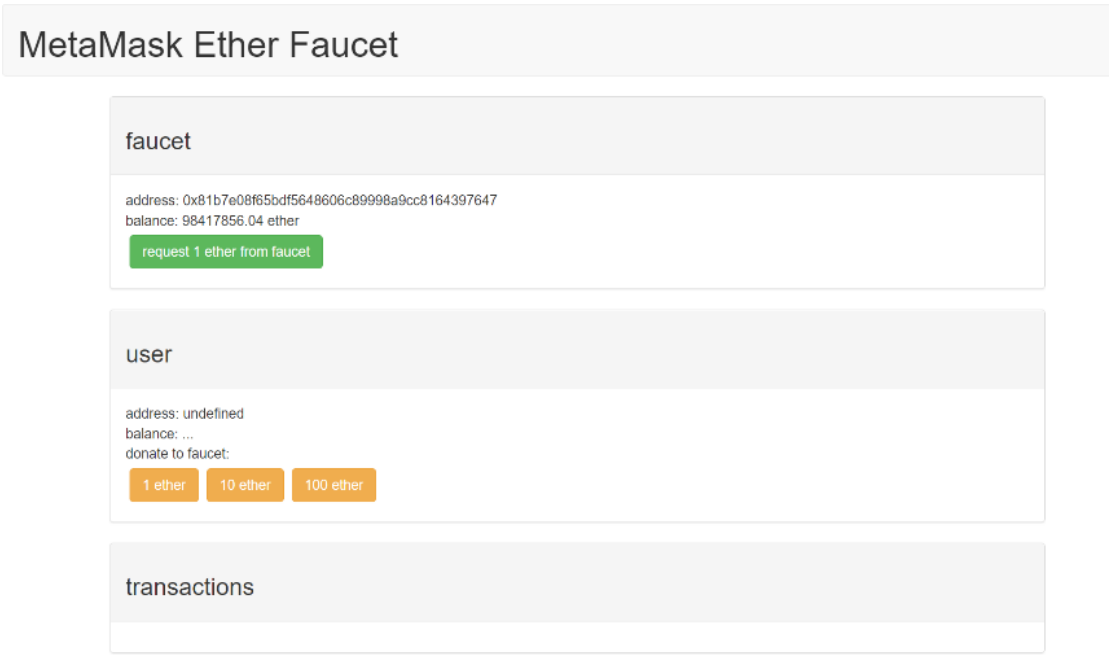
Step 3: MetaMask is ready for deployment. To know more about MetaMask visit the MetaMask official guide.

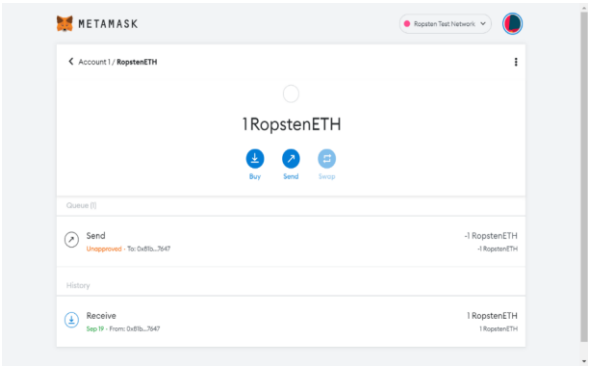
MetaMask comes pre-loaded with fast connections to the Ethereum blockchain and several test networks via our friends at Infura

(opens new window). This allows you to get started without synchronizing a full node, while still providing the option to upgrade your security and use the blockchain provider of your choice.

Today, MetaMask is compatible with any blockchain that exposes an Ethereum-compatible JSON RPC API

(opens new window), including custom and private blockchains. For development, we recommend running a test blockchain like Ganache.





Ether: Ether is the transactional token that facilitates operations on the Ethereum network. All of the programs and services linked with the Ethereum network require computing power, equipment, internet connections, and maintenance. Ether is the payment users give to network participants for executing their requested operations on the network.

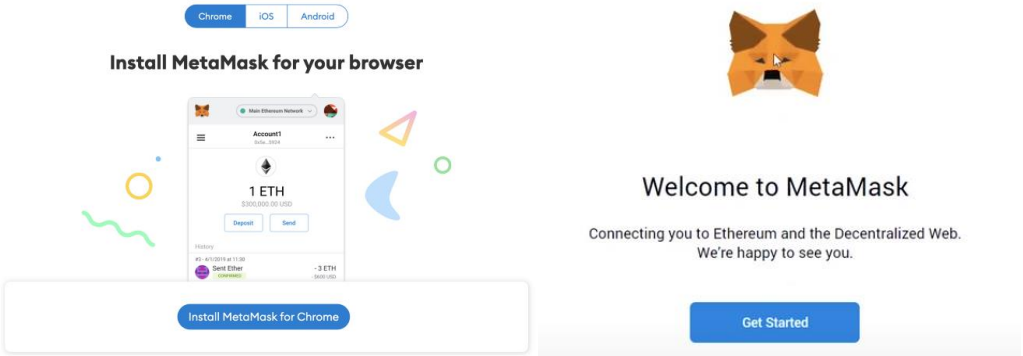
Metaphorically speaking, it is more accurate to refer to ether as the "gas" that powers the network. Gas is the term the community uses to refer to the exchange of ether for the work done to verify transactions and secure the blockchain.

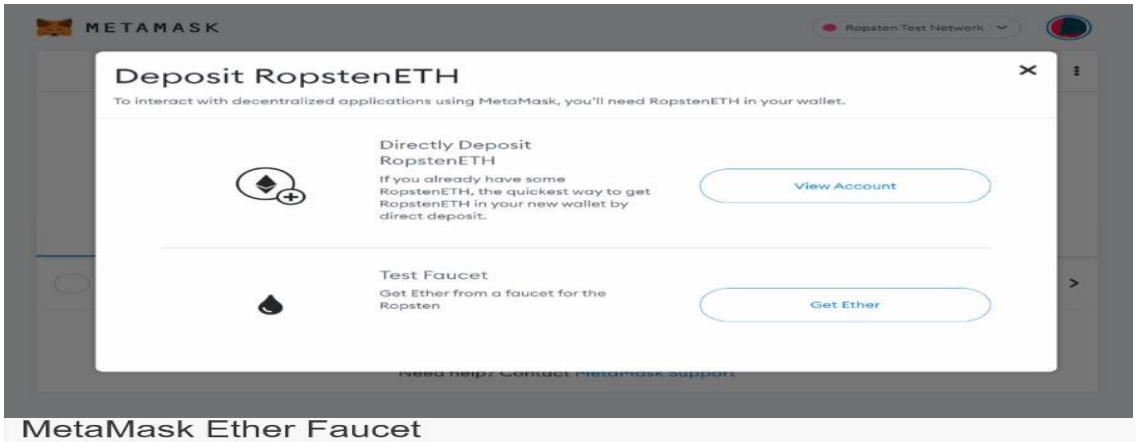
Conclusion: In this assignment we done with metamask and ether spendind per transaction.

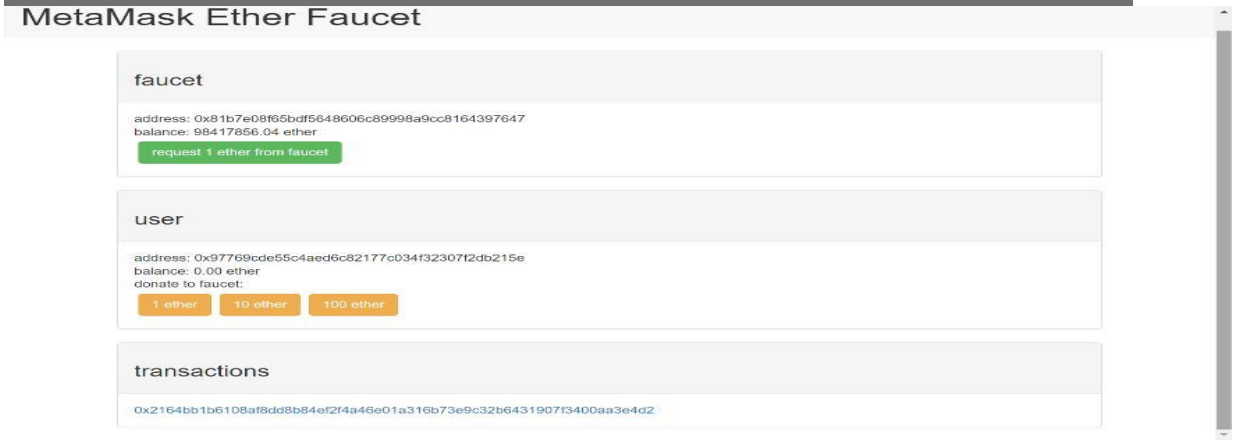
Assignment 2

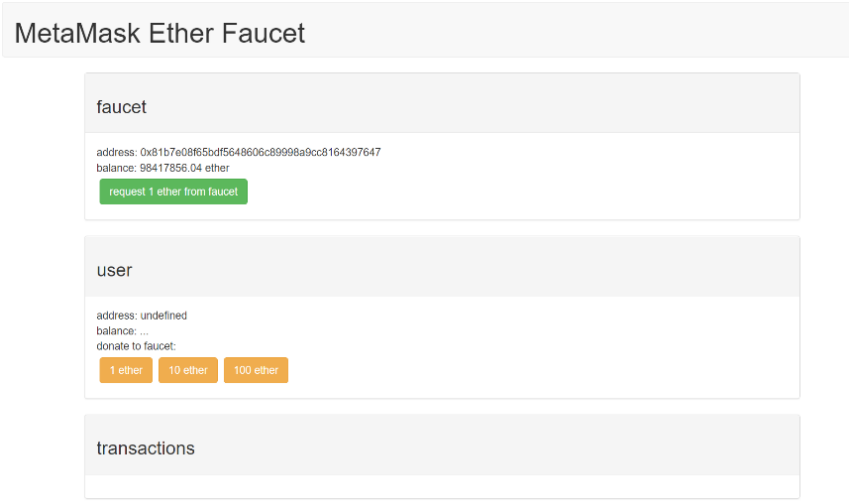
Aim: Create your own wallet using Metamask for crypto transactions.

Create Metamask Wallet.

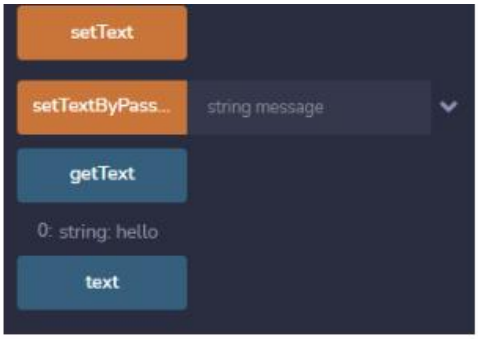




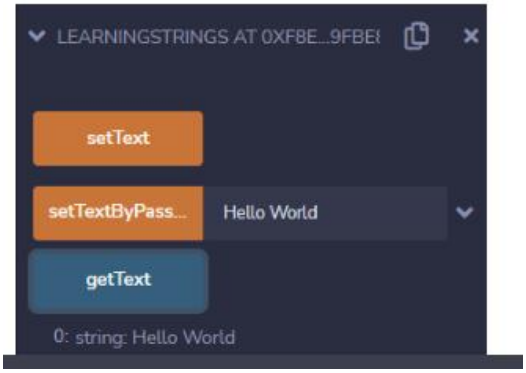





```
// Solidity program to set string value
// using function with parameters
pragma solidity ^0.5.0;

contract LearningStrings
{
        string public text;

        // Assigning the text directly
        function setText() public
        {
                text = 'hello';
        }

        // Assigning the text by passing the value in the function
        function setTextByPassing(string memory message) public
        {
                text = message;
        }

        // Function to get the text
        function getText() view public returns (string memory)
        {
                return text;
        }
}
```



setText()

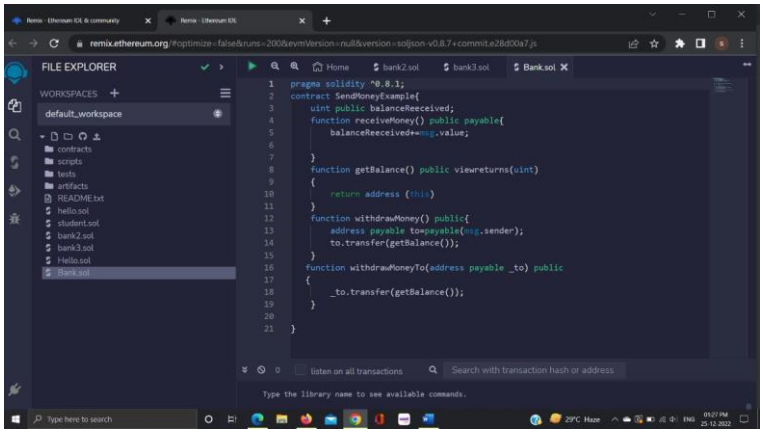


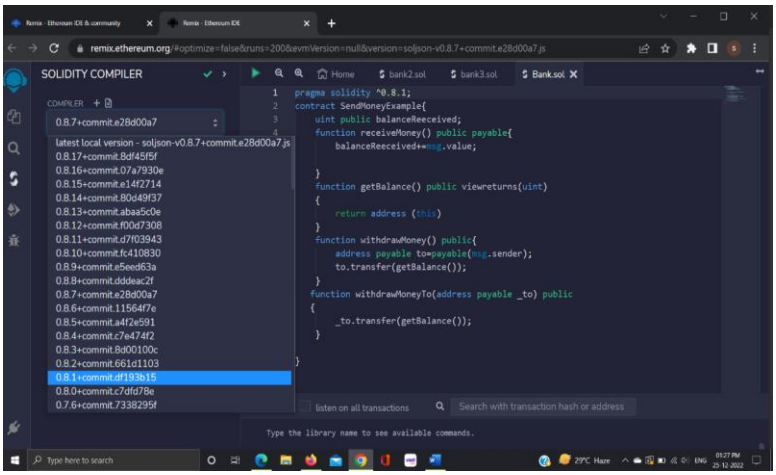
Conclusion: This is how we perform ether per transaction.

Assignment 3

Aim:
Write a smart contract on a test network, for Bank account of a customer for following operations: Deposit money, Withdraw Money , Show balance.

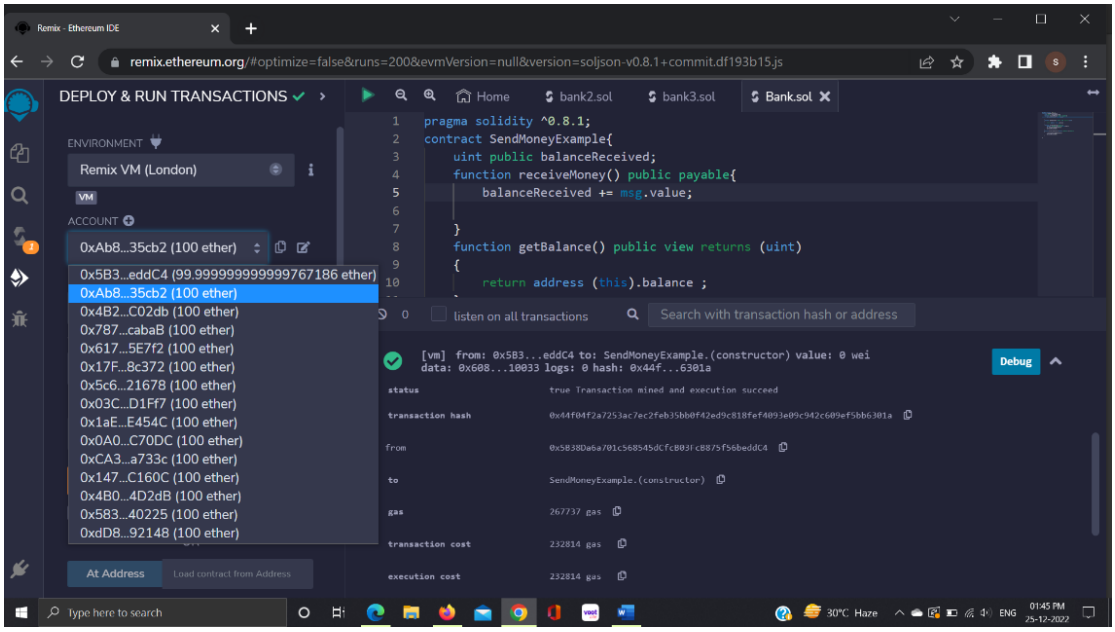
Theory:
Smart contracts are simply programs stored on a block chain that run when predetermined conditions are met. They typically are used to automate the execution of an agreement so that all participants can be immediately certain of the outcome, without any intermediary's involvement or time loss. They can also automate a workflow, triggering the next action when conditions are met. A smart contract is an agreement between two people or entities in the form of computer code programmed to execute automatically



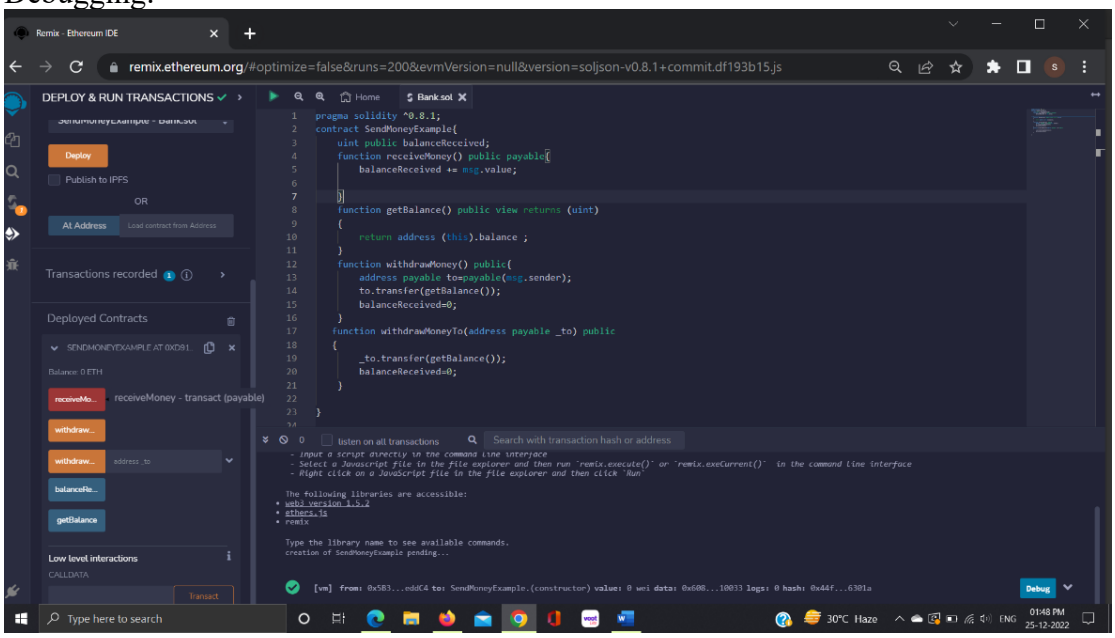


How does a smart contract work:
The operation of a smart contract is similar to other blockchain transfers. These are the necessary steps:
1. A user initiates a transaction from their block chain wallet.
2. The transaction arrives at the distributed database, where the identity is confirmed.
3. The transaction, which may be a transfer of funds, is approved.
4. The transaction includes the code that defines what type of transaction is to be executed.
5. The transactions are added as a block within the block chain.
6. Any change in contract status follows the same process to be updated.

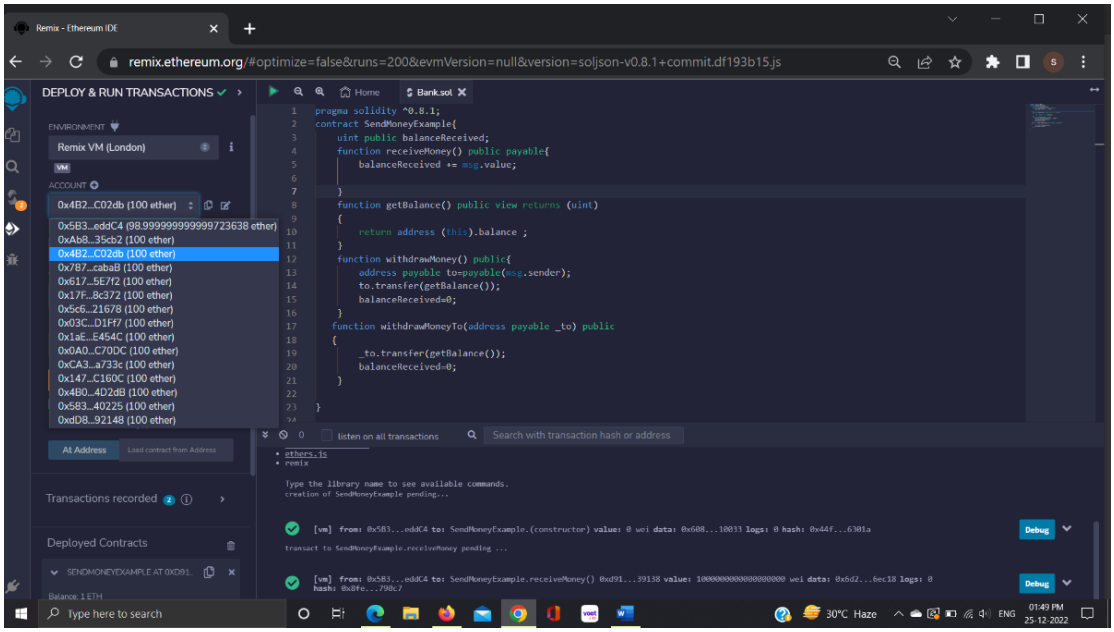
Ether Transaction:

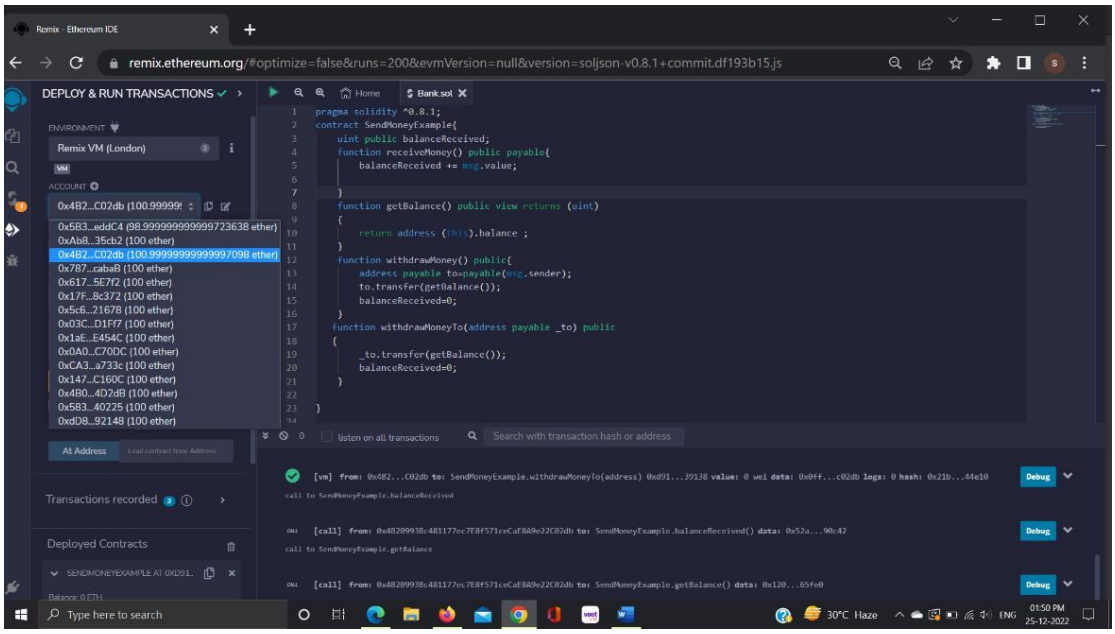

Remix - Ethereum IDE
remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.1+commit.df193b15.js
DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Remix VM (London)
ACCOUNT
0xAb8...35cb2 (100 ether)
0x5B3...eddC4 (99.999999999999767186 ether)
0xAb8...35cb2 (100 ether)
0x4B2...C02db (100 ether)
0x787...cabaB (100 ether)
0x617...5E7f2 (100 ether)
0x17F...8c372 (100 ether)
0x5c6...21678 (100 ether)
0x03C...D1Ff7 (100 ether)
0x1aE...E454C (100 ether)
0x0A0...C70DC (100 ether)
0xCA3...a733c (100 ether)
0x147...C160C (100 ether)
0x4B0...4D2dB (100 ether)
0x583...40225 (100 ether)
0xdD8...92148 (100 ether)
At Address
Load contract from Address
Home
bank2.sol
bank3.sol
Bank.sol
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 ;
listen on all transactions
Search with transaction hash or address
[vm] from: 0x5B3...eddC4 to: SendMoneyExample.(constructor) value: 0 wei
data: 0x608...10033 logs: 0 hash: 0x44f...6301a
Debug
status
true Transaction mined and execution succeed
transaction hash
0x44f04f2a7253ac7ec2feb35bb0f42ed9c818fef4093e09c942c609ef5bb6301a
from
0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to
SendMoneyExample.(constructor)
gas
267737 gas
transaction cost
232814 gas
execution cost
232814 gas
Type here to search
30°C Haze
ENG
01:45 PM
25-12-2022

Debugging:


Remix - Ethereum IDE
remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.1+commit.df193b15.js
DEPLOY & RUN TRANSACTIONS
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded
Deployed Contracts
receiveMoney - transact (payable)
withdraw...
balanceRe...
getBalance
Low level interactions
CALLDATA
Transact
Home
Bank.sol
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 ;
}
function withdrawMoney() public{
address payable to=payable(msg.sender);
to.transfer(getBalance());
balanceReceived=0;
}
function withdrawMoneyTo(address payable _to) public
{
_to.transfer(getBalance());
balanceReceived=0;
}
}
listen on all transactions
Search with transaction hash or address
- Input a script directly in the command line interface
- Select a Javascript file in the file explorer and then run `remix.execute()` or `remix.exeCurrent()` in the command line interface
- Right click on a JavaScript file in the file explorer and then click `Run`
The following libraries are accessible:
web3 version 1.5.2
ethers.js
remix
Type the library name to see available commands.
creation of SendMoneyExample pending...
[vm] from: 0x5B3...eddC4 to: SendMoneyExample.(constructor) value: 0 wei data: 0x608...10033 logs: 0 hash: 0x44f...6301a
Debug
Type here to search
30°C Haze
ENG
01:48 PM
25-12-2022

Ether deduct from one address to another:





Conclusion:

We create a smart contract on a test network, for Bank account of a customer for following
operations: Deposit money ,Withdraw Money, Show balance.

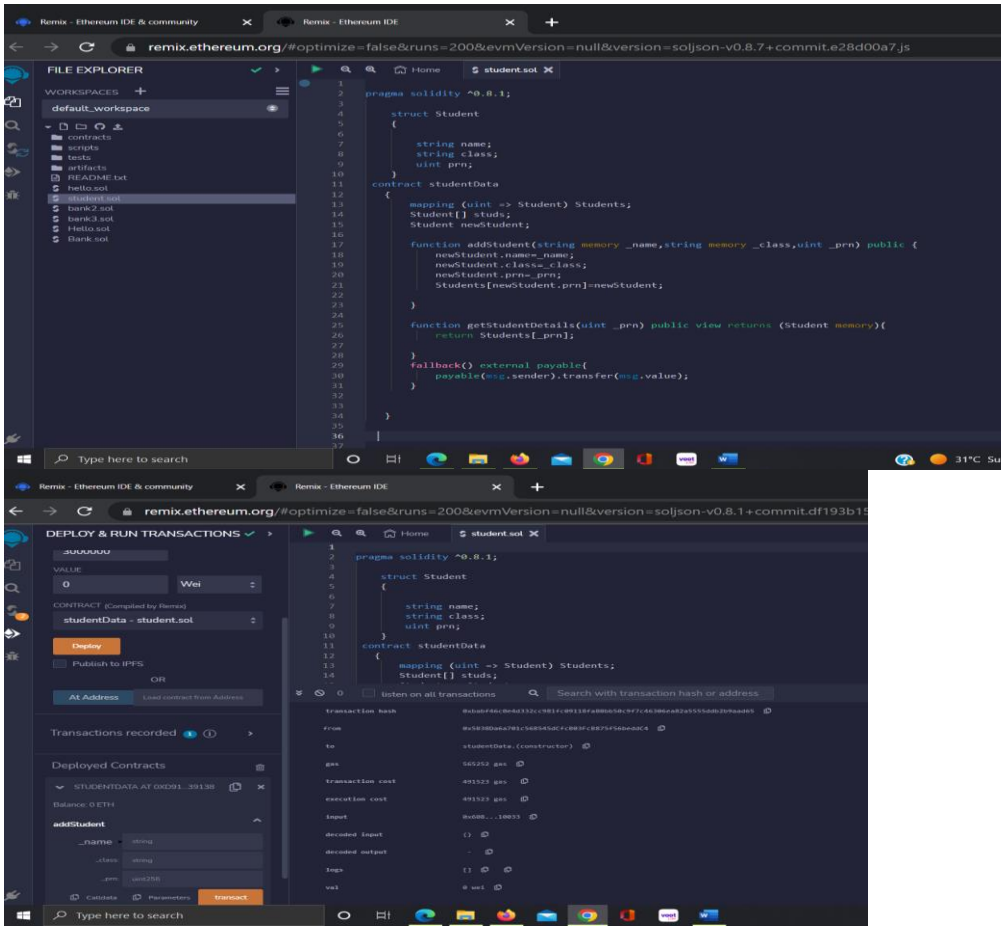
Assignment 4

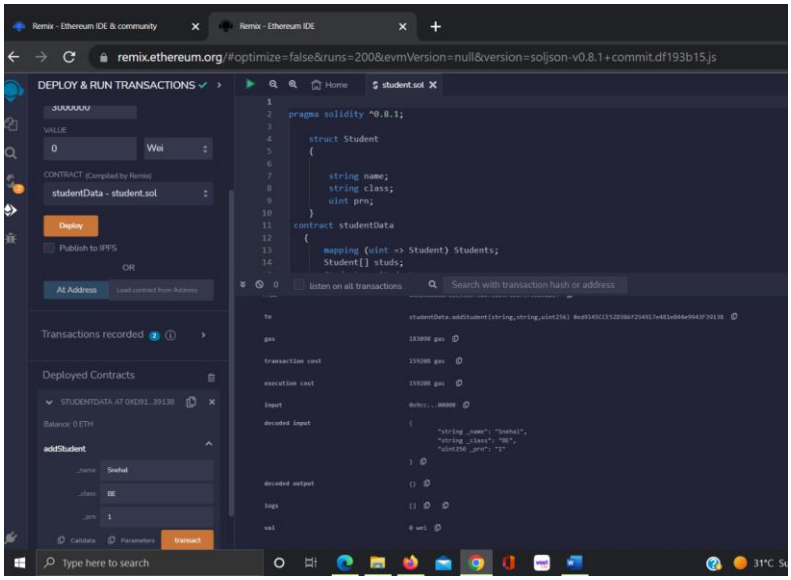
Aim:

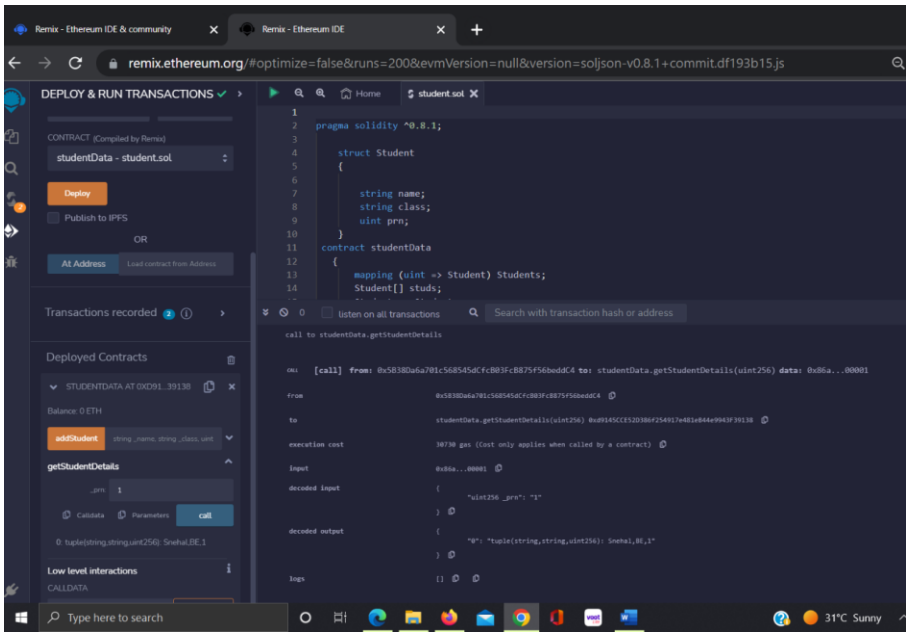
Write a program in solidity to create Student data. Use the following constructs:

- Structures
- Arrays
- Fallback

Deploy this as smart contract on Ethereum and Observe the transaction fee and Gas values.







Fallback Code:

Fallback function BT assignment

```
pragma solidity ^0.4.0;

// Creating a contract
contract fback
{
    // Declaring the state variable
    uint x;

    // Mapping of addresses to their balances
    mapping(address => uint) balance;

    // Creating a constructor
constructor() public
    {
        // value of 10
```

```
        x=10;

    }

    // Creating a function
     function SetX(uint _x) public returns(bool)
    {
        // Set x to the
        // value sent
        x=_x;
        return true;
    }

    // This fallback function
    // will keep all the Ether
    function() public payable
    {
        balance[msg.sender] += msg.value;
    }
}

// Creating the sender contract
contract Sender
{
function transfer() public payable
{
    // Address of Fback contract
    address _receiver =
            0x74348D0905e60909c25533d5BE80Ee2A2194b7a9;

    // Transfers 100 Eth to above contract
    _receiver.transfer(10);
}
}
```

Conclusion:

This is how we create a Student data. Use the following constructs:

- Structures
- Arrays
- Fallback

Deploy this as smart contract on Ethereum and Observe the transaction fee and Gas values.

Assignment 5

Aim: Write a survey report on types of Block chains and its real time use cases.

Theory: The basic application of the block chain is to perform transactions in a secure network. That's why people use block chain and ledger technology in different scenarios. One can set up multi chain to prevent unauthorized access to sensitive data. It is not available to the public, and can only be available to authorized entities in the organization. It depends on the organization which type it requires to choose for their By using block chain we can track orders and payments from end to end.

Advantage using blockchain :

It provides greater security among data. It provides greater trust among users. Reduce the cost of production. Improve Speed. Invocation and tokenization. It provides immutable records.

Disadvantages using blockchain :

Data modification is not possible. It requires large storage for a large database. The owner cannot access the private key again if they forget or lose it.Real life application of blockchain. Here is a list of real world problem where we can use blockchain. In a secure and full-proof voting management system. To supply chain management. In healthcare management. Real estate project.NFT marketplace. Avoid copyright and original content creation. In the personal identity system .To make an immutable data backup. Internet of Things

Permissionless Blockchain

It is also known as trustless or public blockchains, are available to everyone to participate in the blockchains process that use to validate transactions and data. These are used in the network where high transparency is required.

Characteristics:

- Permissionless blockchain has no central authority.
- The platform is completely open-source.
- Full transparency of the transaction.
- Heavy use of tokens.

Advantages:

- Everyone can participate only requirement is good hardware and internet.
- Bring trust among users or entities.
- It has a high level of transparency as it's a larger network.
- Broader decentralization of access to more participants.

Disadvantages:

- Poor energy efficiency due to large network.
- Lower performance scalability.

- Less privacy as many of the things is visible.

Permissioned Blockchain

These are the closed network only a set of groups are allowed to validate transactions or data in a given blockchain network. These are used in the network where high privacy and security are required.

Characteristics:

- A major feature is a transparency based on the objective of the organization.
- Another feature is the lack of anatomy as only a limited number of users are allowed.
- It does not have a central authority.
- Developed by private authority.

Advantages:

- This blockchain tends to be faster as it has some nodes for validations.
- They can offer customizability.
- Strong Privacy as permission is needed for accessing transaction information.
- As few nodes are involved performance and scalability are increased.

Disadvantages:

- Not truly decentralized as it requires permission
- Risk of corruption as only a few participants are involved.
- Anytime owner and operator can change the rules as per their need.

1. Public Blockchain

These blockchains are completely open to following the idea of decentralization. They don't have any restrictions, anyone having a computer and internet can participate in the network.

- As the name is public this blockchain is open to the public, which means it is not owned by anyone.
- Anyone having internet and a computer with good hardware can participate in this public blockchain.
- All the computer in the network hold the copy of other nodes or block present in the network
- In this public blockchain, we can also perform verification of transactions or records

Advantages:

- **Trustable:** There are algorithms to detect no fraud. Participants need not worry about the other nodes in the network
- **Secure:** This blockchain is large in size as it is open to the public. In a large size, there is greater distribution of records
- **Anonymous Nature:** It is a secure platform to make your transaction properly at the same time, you are not required to reveal your name and identity in order to participate.

- **Decentralized:** There is no single platform that maintains the network, instead every user has a copy of the ledger.

Disadvantages:

- **Processing:** The rate of the transaction process is very slow, due to its large size. Verification of each node is a very time-consuming process.
- **Energy Consumption:** Proof of work is high energy-consuming. It requires good computer hardware to participate in the network
- **Acceptance:** No central authority is there so governments are facing the issue to implement the technology faster.

Use Cases: Public Blockchain is secured with proof of work or proof of stake they can be used to displace traditional financial systems. The more advanced side of this blockchain is the smart contract that enabled this blockchain to support decentralization. Examples of public blockchain are Bitcoin, Ethereum.

2. Private Blockchain

These blockchains are not as decentralized as the public blockchain only selected nodes can participate in the process, making it more secure than the others.

- These are not as open as a public blockchain.
- They are open to some authorized users only.
- These blockchains are operated in a closed network.
- In this few people are allowed to participate in a network within a company/organization.

Advantages:

- **Speed:** The rate of the transaction is high, due to its small size. Verification of each node is less time-consuming.
- **Scalability:** We can modify the scalability. The size of the network can be decided manually.
- **Privacy:** It has increased the level of privacy for confidentiality reasons as the businesses required.
- **Balanced:** It is more balanced as only some user has the access to the transaction which improves the performance of the network.

Disadvantages:

- Security- The number of nodes in this type is limited so chances of manipulation are there. These blockchains are more vulnerable.
- Centralized- Trust building is one of the main disadvantages due to its central nature. Organizations can use this for malpractices.
- Count- Since there are few nodes if nodes go offline the entire system of blockchain can be endangered.

Use Cases: With proper security and maintenance, this blockchain is a great asset to secure information without exposing it to the public eye. Therefore companies use them for internal auditing, voting, and asset management. An example of private blockchains is Hyperledger, Corda.

3. Hybrid Blockchain

It is the mixed content of the private and public blockchain, where some part is controlled by some organization and other makes are made visible as a public blockchain.

- It is a combination of both public and private blockchain.

- Permission-based and permissionless systems are used.
- User access information via smart contracts
- Even a primary entity owns a hybrid blockchain it cannot alter the transaction

Advantages:

- **Ecosystem:** Most advantageous thing about this blockchain is its hybrid nature. It cannot be hacked as 51% of users don't have access to the network
- **Cost:** Transactions are cheap as only a few nodes verify the transaction. All the nodes don't carry the verification hence less computational cost.
- **Architecture:** It is highly customizable and still maintains integrity, security, and transparency.
- **Operations:** It can choose the participants in the blockchain and decide which transaction can be made public.

Disadvantages:

- **Efficiency:** Not everyone is in the position to implement a hybrid Blockchain. The organization also faces some difficulty in terms of efficiency in maintenance.
- **Transparency:** There is a possibility that someone can hide information from the user. If someone wants to get access through a hybrid blockchain it depends on the organization whether they will give or not.
- **Ecosystem:** Due to its closed ecosystem this blockchain lacks the incentives for network participation.

Use Case: It provides a greater solution to the health care industry, government, real estate, and financial companies. It provides a remedy where data is to be accessed publicly but needs to be shielded privately. Examples of Hybrid Blockchain are Ripple network and XRP token.

4. Consortium Blockchain

It is a creative approach that solves the needs of the organization. This blockchain validates the transaction and also initiates or receives transactions.

- Also known as Federated Blockchain.
- This is an innovative method to solve the organization's needs.
- Some part is public and some part is private.
- In this type, more than one organization manages the blockchain.

Advantages:

- **Speed:** A limited number of users make verification fast. The high speed makes this more usable for organizations.
- **Authority:** Multiple organizations can take part and make it decentralized at every level. Decentralized authority, makes it more secure.
- **Privacy:** The information of the checked blocks is unknown to the public view. but any member belonging to the blockchain can access it.
- **Flexible:** There is much divergence in the flexibility of the blockchain. Since it is not a very large decision can be taken faster.

Disadvantages:

- **Approval:** All the members approve the protocol making it less flexible. Since one or more organizations are involved there can be differences in the vision of interest.

- **Transparency:** It can be hacked if the organization becomes corrupt. Organizations may hide information from the users.
- **Vulnerability:** If few nodes are getting compromised there is a greater chance of vulnerability in this blockchain

Use Cases: It has high potential in businesses, banks, and other payment processors. Food tracking of the organizations frequently collaborates with their sectors making it a federated solution ideal for their use. Examples of consortium Blockchain are Tendermint and Multichain.

Conclusion: This is how we study real time cases of blockchain.

Assignment 6

Aim: Write a program to create a Business Network using Hyperledger.

Theory: Create Business Network Using Hyperledger Composer

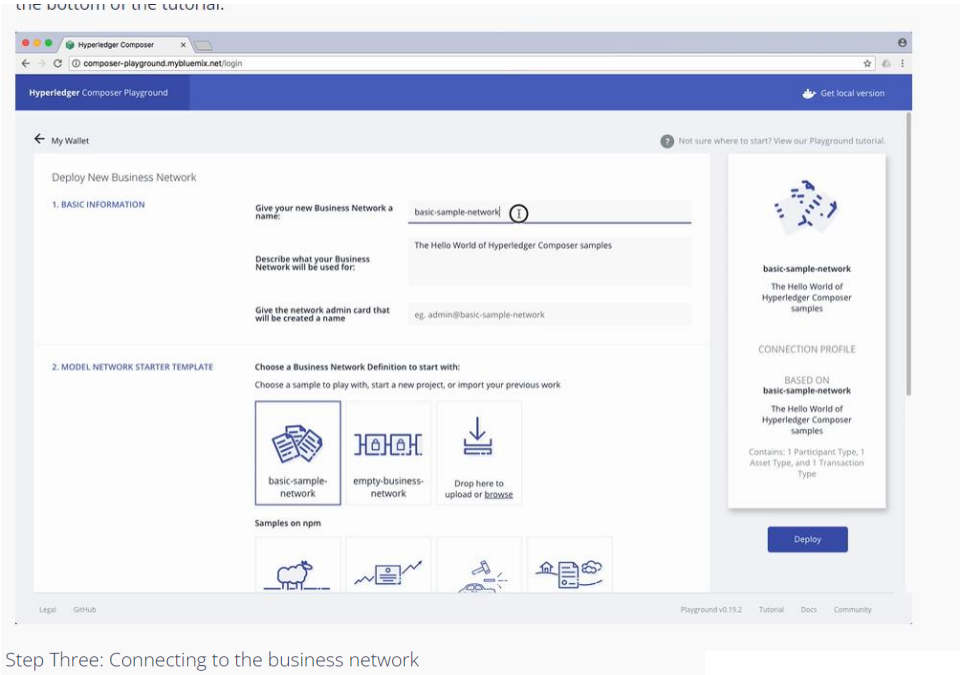
Open Composer Playground (note, this link will take you to the web Composer Playground - you can also follow along in a local version if you've already installed the development environment).

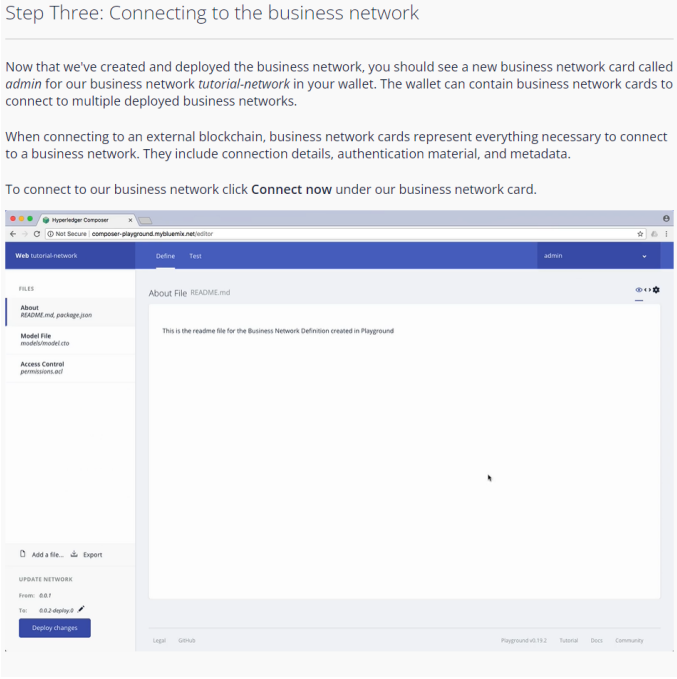
You should see the My Business Networks screen. The My Business Networks page shows you a summary of the business networks you can connect to, and the identities you can use to connect to them. Don't worry about this too much for the time being, as we're going to create our own network.

Creating a new business network

Next, we want to create a new business network from scratch. A business network has a couple of defining properties; a name, and an optional description. You can also choose to base a new business network on an existing template, or import your own template.

1. Click Deploy a new business network under the Web Browser heading to get started.
2. The new business network needs a name, let's call it `tutorial-network`.
3. Optionally, you can enter a description for your business network.
4. Next we must select a business network to base ours on, because we want to build the network from scratch, click empty-business-network.
5. Now that our network is defined, click Deploy.





As you can see, we're in the Define tab right now, this tab is where you create and edit the files that make up a business network definition, before deploying them and testing them using the Test tab.

As we selected an empty business network template, we need to modify the template files provided. The first step is to update the model file. Model files define the assets, participants, transactions, and events in our business network.

For more information on our modeling language, check our documentation.

Click the Model file to view it.

1. Delete the lines of code in the model file and replace it with this:

```
/**
 * My commodity trading network
 */
namespace org.example.mynetwork
asset Commodity identified by tradingSymbol {
    o String tradingSymbol
    o String description
    o String mainExchange
    o Double quantity
    --> Trader owner
}
participant Trader identified by tradeId {
    o String tradeId
    o String firstName
    o String lastName
}
transaction Trade {
    --> Commodity commodity
    --> Trader newOwner
}
```

This domain model defines a single asset type `Commodity` and single participant type `Trader` and a single transaction type `Trade` that is used to modify the owner of a commodity.

Now that the domain model has been defined, we can define the transaction logic for the business network. Composer expresses the logic for a business network using JavaScript functions. These functions are automatically executed when a transaction is submitted for processing.

For more information on writing transaction processor functions, check our documentation.

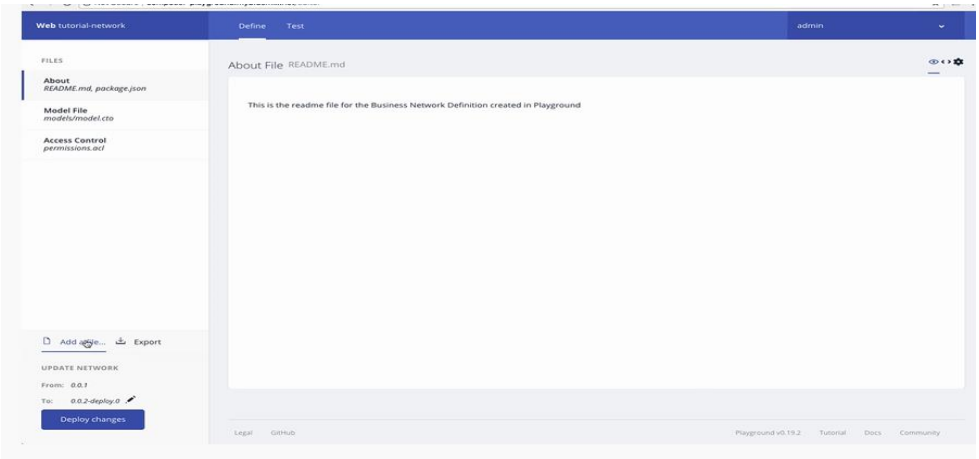
1. Click the Add a file button.
2. Click the Script file and click Add.

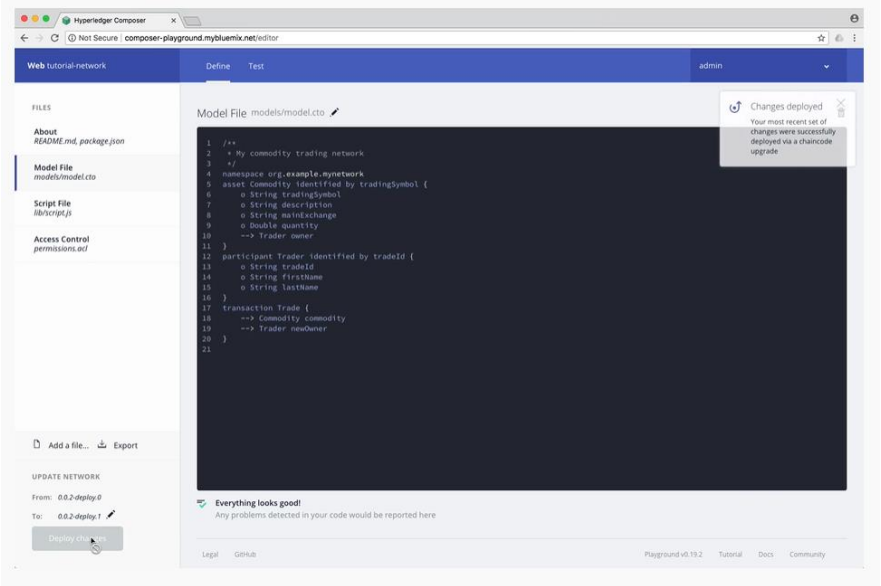
3. Delete the lines of code in the script file and replace it with the following code:

Copy

```
/**

 * Track the trade of a commodity from one trader to another

 * @param {org.example.mynetwork.Trade} trade - the trade to be
processed

 * @transaction

 */

asyncfunctiontradeCommodity(trade){

trade.commodity.owner=trade.newOwner;

letassetRegistry=awaitgetAssetRegistry('org.example.mynetwork.C
ommodity');

awaitassetRegistry.update(trade.commodity);

}
```

This function simply changes the `owner` property on a commodity based on the `new Owner` property on an incoming `Trade` transaction. It then persists the modified `Commodity` back into the asset registry, used to store `Commodity` instances.





The first thing we should add to our business network is two participants.

1. Ensure that you have the Trader tab selected on the left, and click Create New Participant in the upper right.
2. What you can see is the data structure of a *Trader* participant. We want some easily recognizable data, so delete the code that's there and paste the following:

Copy

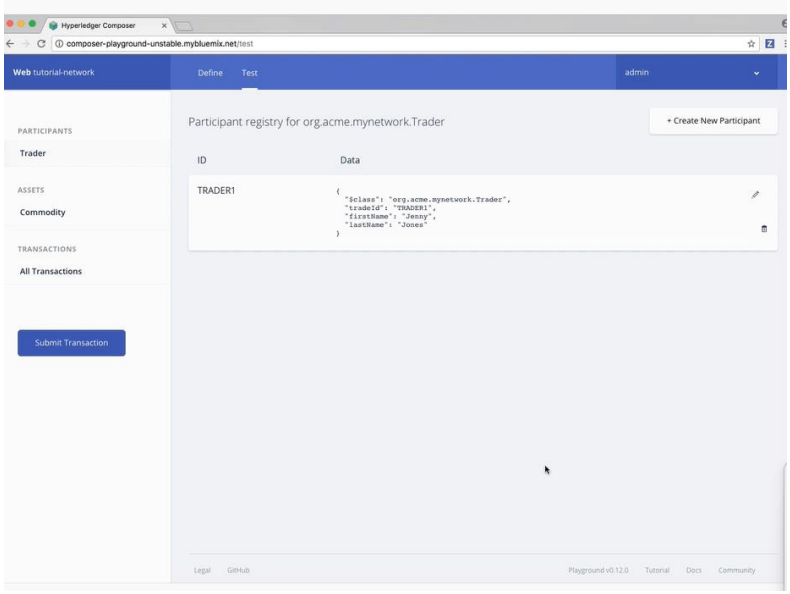
```
{
  "$class": "org.example.mynetwork.Trader",
  "tradeId": "TRADER1",
  "firstName": "Jenny",
  "lastName": "Jones"
}
```

3. Click Create New to create the participant.
4. You should be able to see the new Trader participant you've created. We need another Trader to test our Trade transaction though, so create another Trader, but this time, use the following data:

Copy

```
{
  "$class": "org.example.mynetwork.Trader",
  "tradeId": "TRADER2",
```

```
  "firstName": "Amy",

  "lastName": "Williams"

}
```



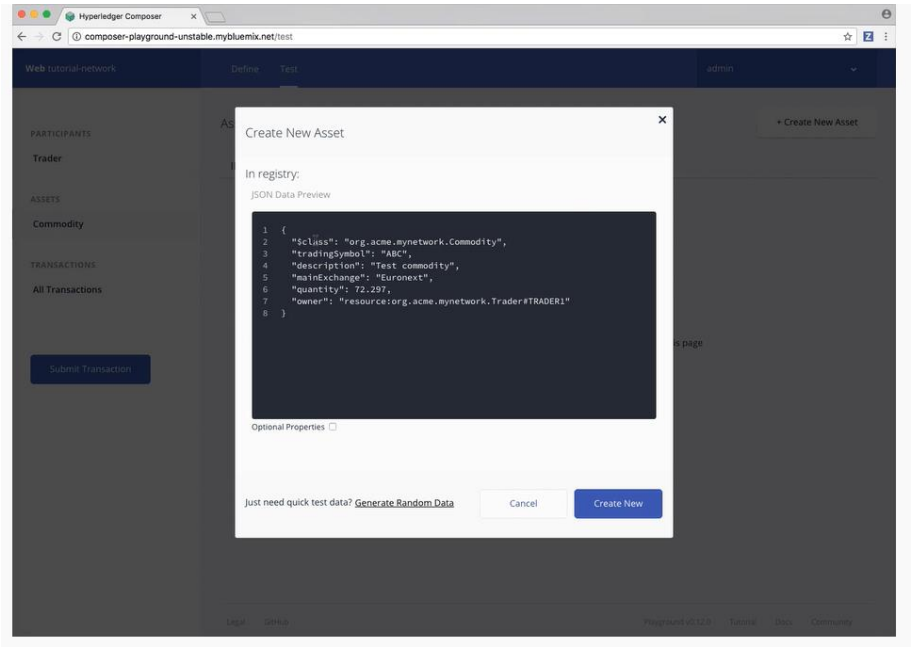
Now that we have two Trader participants, we need something for them to trade. Creating an asset is very similar to creating a participant. The Commodity we're creating will have an owner property indicating that it belongs to the Trader with the trade Id of `TRADER1`.

1. Click the Commodity tab under Assets and click Create New Asset.
2. Delete the asset data and replace it with the following:

Copy

```
{

  "$class": "org.example.mynetwork.Commodity",

  "tradingSymbol": "ABC",

  "description": "Test commodity",

  "mainExchange": "Euronext",

  "quantity": 72.297,

  "owner": "resource:org.example.mynetwork.Trader#TRADER1"

}
```

3. After creating this asset, you should be able to see it in the Commodity tab.



.To test the Trade transaction: Click the Submit Transaction button on the left.Ensure that the transaction type is Trade. Replace the transaction data with the following, or just change the details:

```
{
  "$class": "org.example.mynetwork.Trade",
  "commodity": "resource:org.example.mynetwork.Commodity#ABC",
  "newOwner": "resource:org.example.mynetwork.Trader#TRADER2"
}
```

Click Submit.

Conclusion: This is how we create business network on hyper ledger.