Practical No: 1

Aim:

Installation of metamask and study ether spending per transaction.

Objective:

1. To understand metamask.
2. To understand how the transaction works on Ethereum.

Requirement:

1. Web browser
2. Metamask extension
3. ganache

Theory:

Metamask:

Metamask is a popular cryptocurrency wallet known for its ease of use, availability on both desktops and mobile devices, the ability to buy, send, and receive cryptocurrency from within the wallet, and collect non-fungible tokens across two blockchains.

Metamask allows users to store and manage account keys, broadcast transactions, send and receive Ethereum-based cryptocurrencies and tokens, and securely connect to decentralized applications through a compatible web browser or the mobile app's built-in browser. Websites or other decentralized applications are able to connect, authenticate, and integrate other smart contract functionality with a user's Metamask wallet via JavaScript code that allows the website to send action prompts, signature requests, or transaction requests to the user through Metamask as an intermediary. The application includes an integrated service for exchanging Ethereum tokens by aggregating several decentralized exchanges to find the best exchange rate. This feature, branded as Metamask Swaps, charges a service fee of 0.875% of the transaction amount.

Ethereum:

Ethereum is a decentralized, open-source blockchain with smart contract functionality. Ether is the native cryptocurrency of the platform. Among cryptocurrencies, ether is second only to bitcoin in market capitalization. Ethereum was conceived in 2013

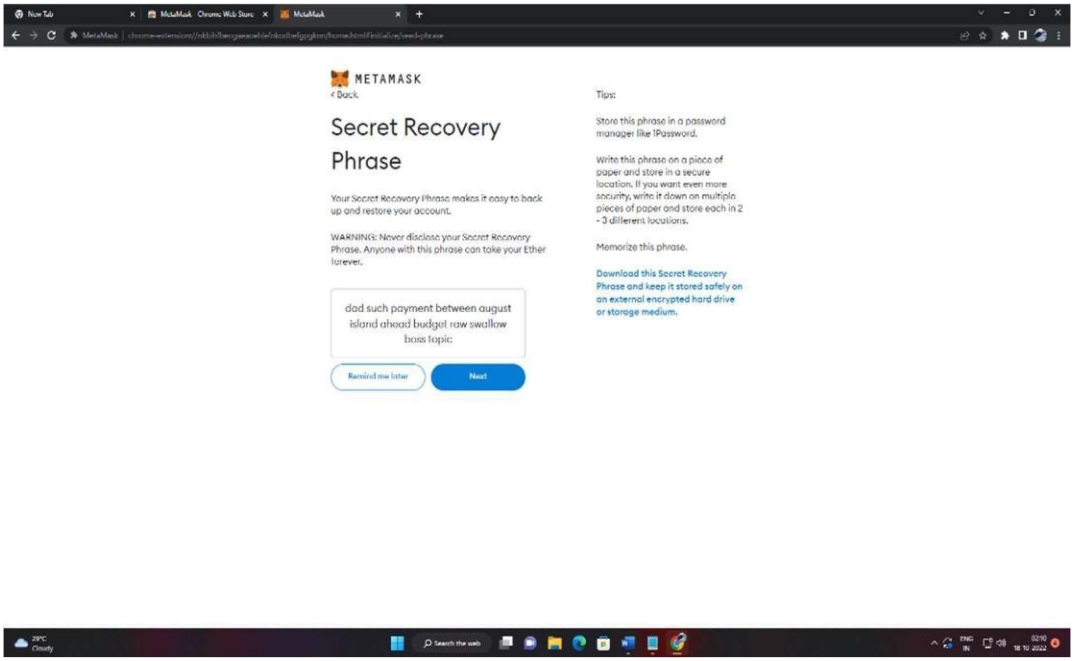
by programmer Vitalik Buterin. Additional founders of Ethereum included Gavin Wood,Charles Hoskinson, Anthony Di Iorio and Joseph Lubin. In 2014, development work began and was crowdfunded, and the network went live on 30 July 2015. Ethereum allows anyone to deploy permanent and immutable decentralized applications onto it, with which users can interact. Decentralized finance applications provide a broad array of financial services without the need for typical financial intermediaries like brokerages, exchanges, or banks, such as allowing cryptocurrency users to borrow against their holdings or lend them out for interest. Ethereum also allows users to create and exchange NFTs, which are unique tokens representing ownership of an associated asset or privilege, as recognized by any number of institutions. Additionally, many other cryptocurrencies utilize the ERC- 20 token standard on top of the Ethereum blockchain and have utilized the platform for initial coin offerings.

Ether:

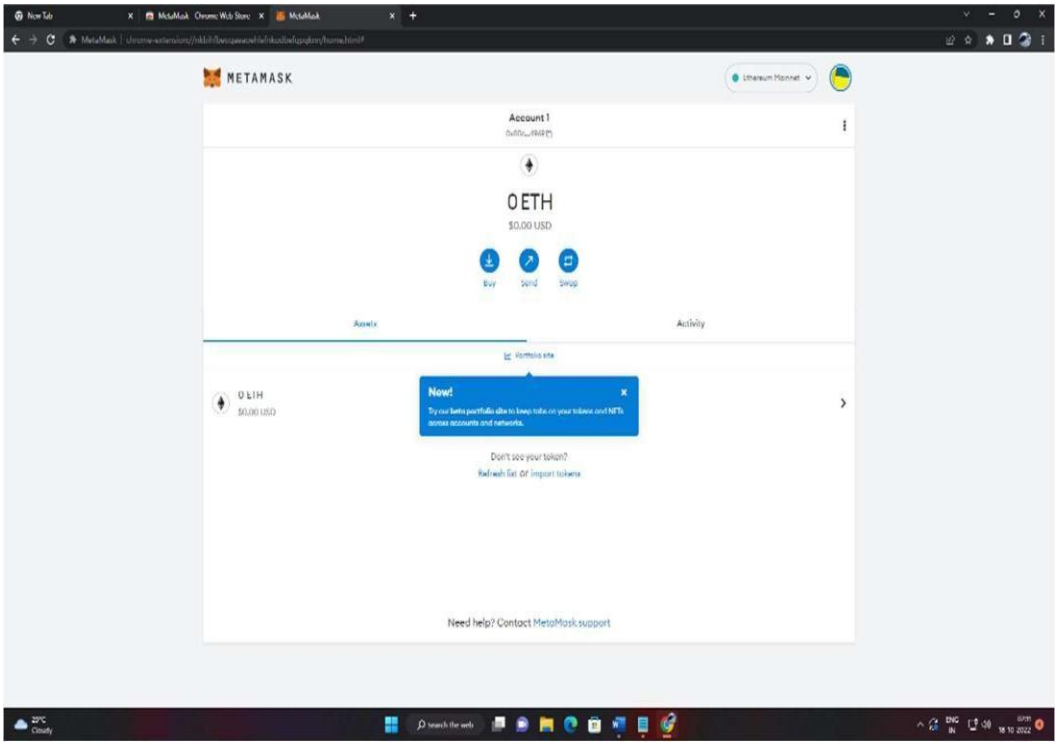
Ether is the transactional token that facilitates operations on the Ethereum network. All of the programs and services linked with the Ethereum network require computing power, equipment, internet connections, and maintenance. Ether is the payment users give to network participants for executing their 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. On 15 September 2022, Ethereum transitioned its consensus mechanism from proof-of-work to proof-ofstake in an upgrade process known as "the Merge". Consequently, Ethereum's energy consumption rate was reduced by about 99.95%, translating to an estimated 110 of annual energy savings. The upgrade required no actions from Ethereum's users. During the Merge, Ethereum's proof-of-work mining mechanism was replaced with the Beacon Chain, a proof-of-stake blockchain network secured by validators staking ether.

STEPS:

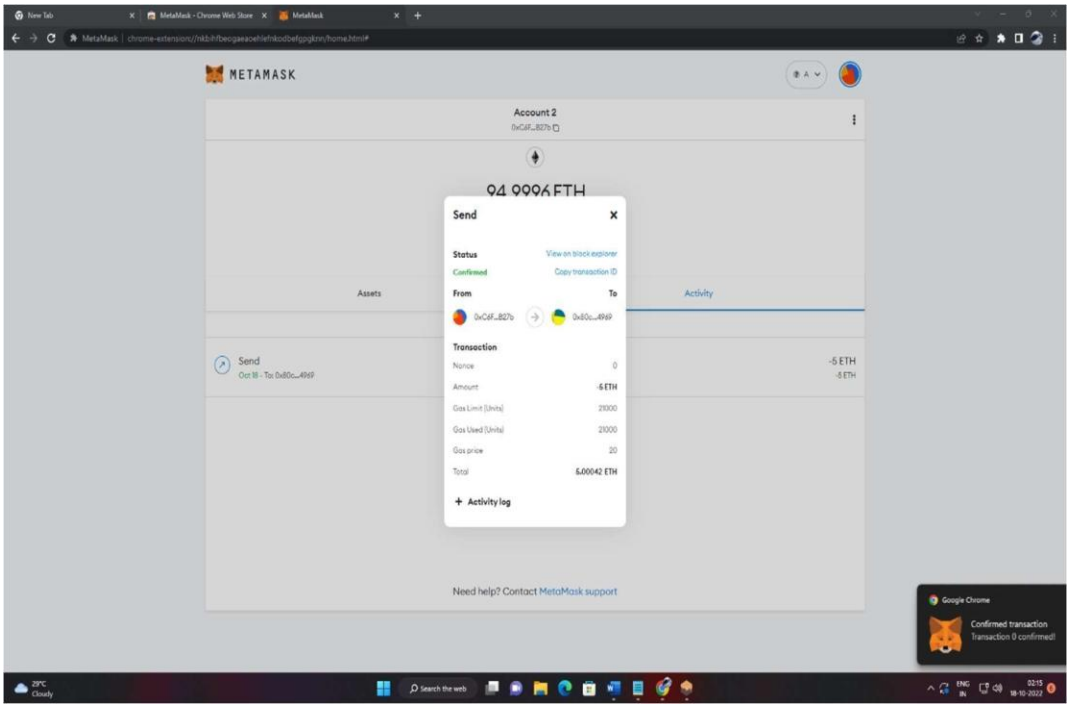
Installation of Metamask:



First of step of metamask is that it requires these kinds of steps which requires to enter key passwords.



This is the interface for the metamask, which shows the account details.



This is a transaction details for ethers on Ethereum network.

Conclusion: We learned about the installation of metamask. We also learned how the transaction works on metamask.

ASSESMENT:

Exp. No. 01	**Roll No.**		**Date of starting**	
			Date of completion	
Remarks/ Grade				**Signature of subject teacher**

Practical No. 2

Aim:

Create your own wallet using Metamask for crypto transactions.

Objective:

1. To understand meta wallet.
2. To understand how the crypto transaction works.

Requirement:

1. Web browser
2. Metamask extension.
3. Ganache

Theory:

Cryptocurrency:

Crypto-currency, or crypto is a digital currency designed to work as a medium of exchange through a computer network that is not reliant on any central authority, such as a government or bank, to uphold or maintain it. It is a decentralized system for verifying that the parties to a transaction have the money they claim to have, eliminating the need for traditional intermediaries, such as banks, when funds are being transferred between two entities. Individual coin ownership records are stored in a digital ledger, which is a computerized database using strong cryptography to secure transaction records, to control the creation of additional coins, and to verify the transfer of coin ownership. Despite their name, cryptocurrencies are not considered to be currencies in the traditional sense and while varying treatments have been applied to them, including classification as commodities, securities, as well as currencies, cryptocurrencies are generally viewed as a distinct asset class in practice. Some crypto schemes use validators to maintain the cryptocurrency. In a proof-of-stake model, owners put up their tokens as collateral. In return, they get authority over the token in proportion to the amount they stake. Cryptocurrencies typically use decentralized control as opposed to a central bank digital currency. When a cryptocurrency is minted or created prior to issuance or issued by a single issuer, it is generally considered centralized. When implemented with decentralized

control, each cryptocurrency works through distributed ledger technology, typically a blockchain, that serves as a public financial transaction database. Traditional asset classes like currencies, commodities, and stocks, as well as macroeconomic factors, have modest exposures to cryptocurrency returns. The first decentralized cryptocurrency was Bitcoin, which first released as open-source software in 2009. As of March 2022, there were more than 9,000 other cryptocurrencies in the marketplace, of which more than 70 had a market capitalization exceeding $1 billion.

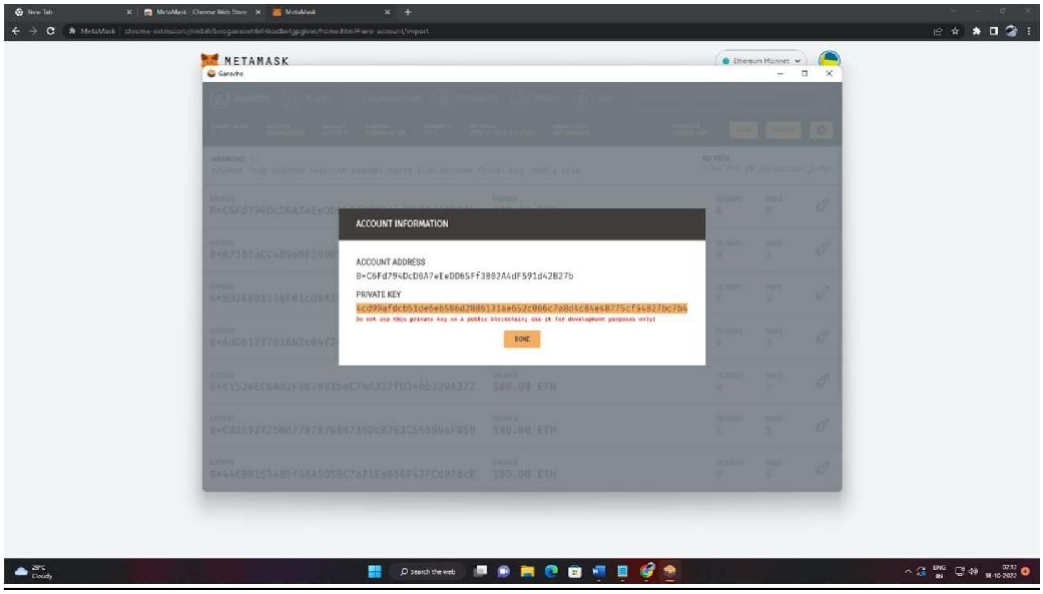
Crypto Wallet:

Cryptocurrency wallets store users' public and private keys while providing an easy-to-use interface to manage crypto balances. They also support cryptocurrency transfers through the blockchain. Some wallets even allow users to perform certain actions with their crypto assets such as buying and selling or interacting with decentralized applications.

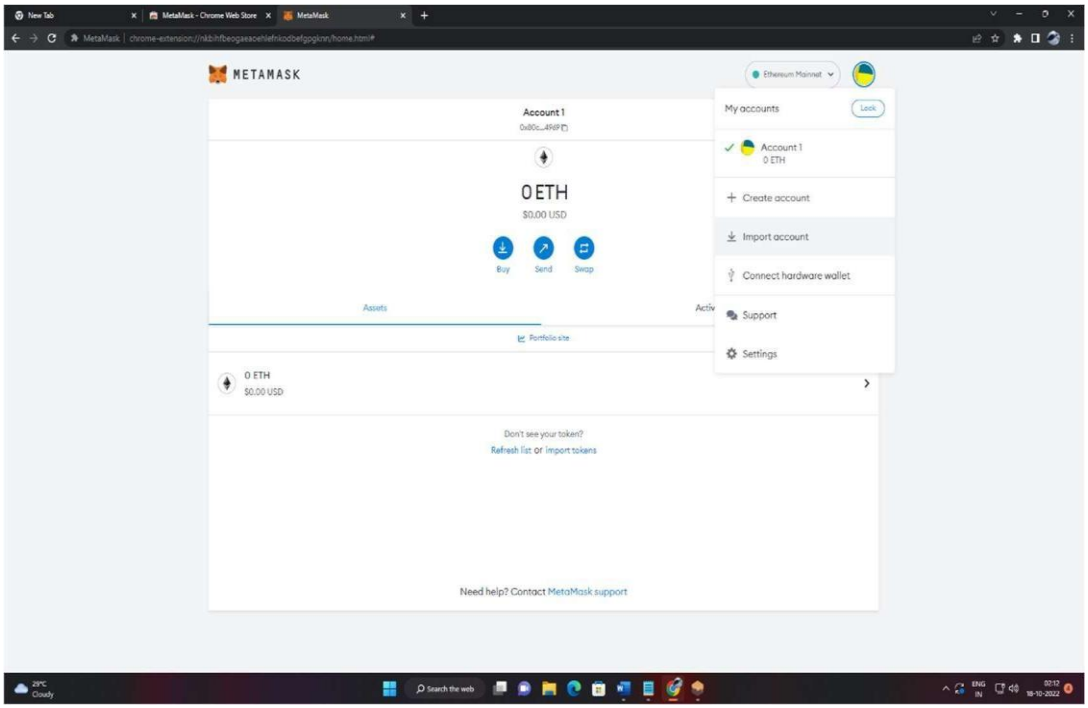
It is important to remember that cryptocurrency transactions do not represent a 'sending' of crypto tokens from your mobile phone to someone else's mobile phone. When you are sending tokens, you are actually using your private key to sign the transaction and broadcast it to the blockchain network. The network will then include your transaction to reflect the updated balance in your address and the recipient's.

So, the term 'wallet' is actually somewhat of a misnomer as crypto wallets don't really store cryptocurrency in the same way physical wallets hold cash. Instead, they read the public ledger to show you the balances in your addresses and also hold the private keys that enable you to make transactions.

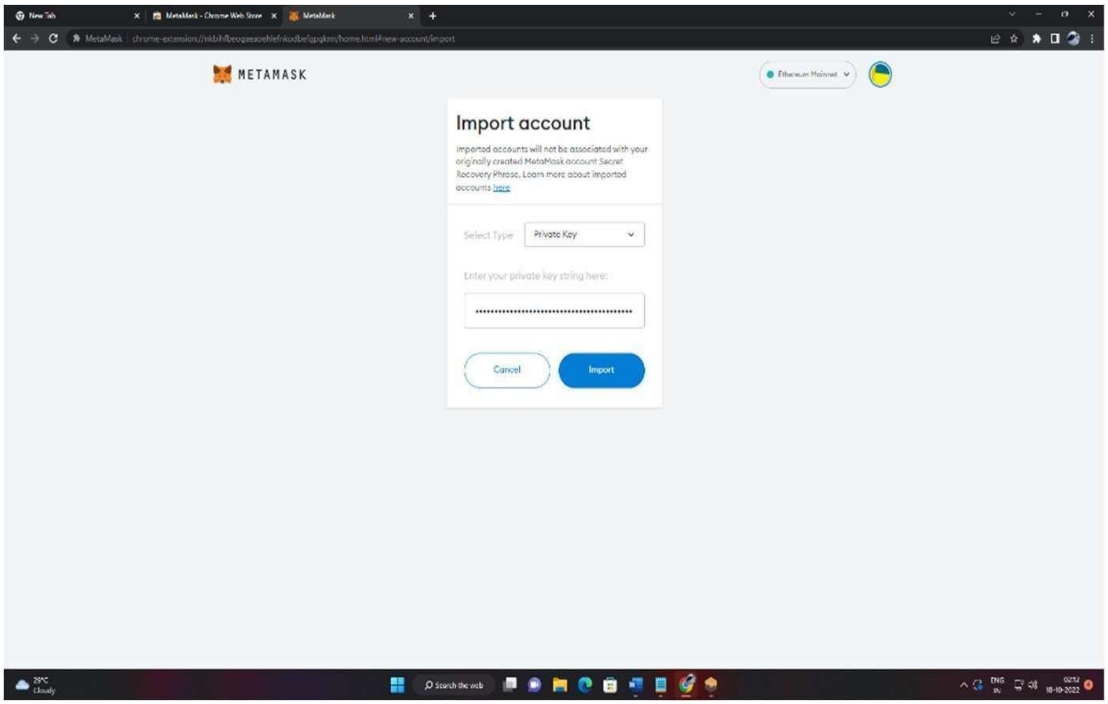
STEPS:



We are importing account form the ganache as our crypto wallet. This wallet contains local ethers with which we are using to create our crypto wallet. With the help of the private key, we are importing account into our crypto wallet.

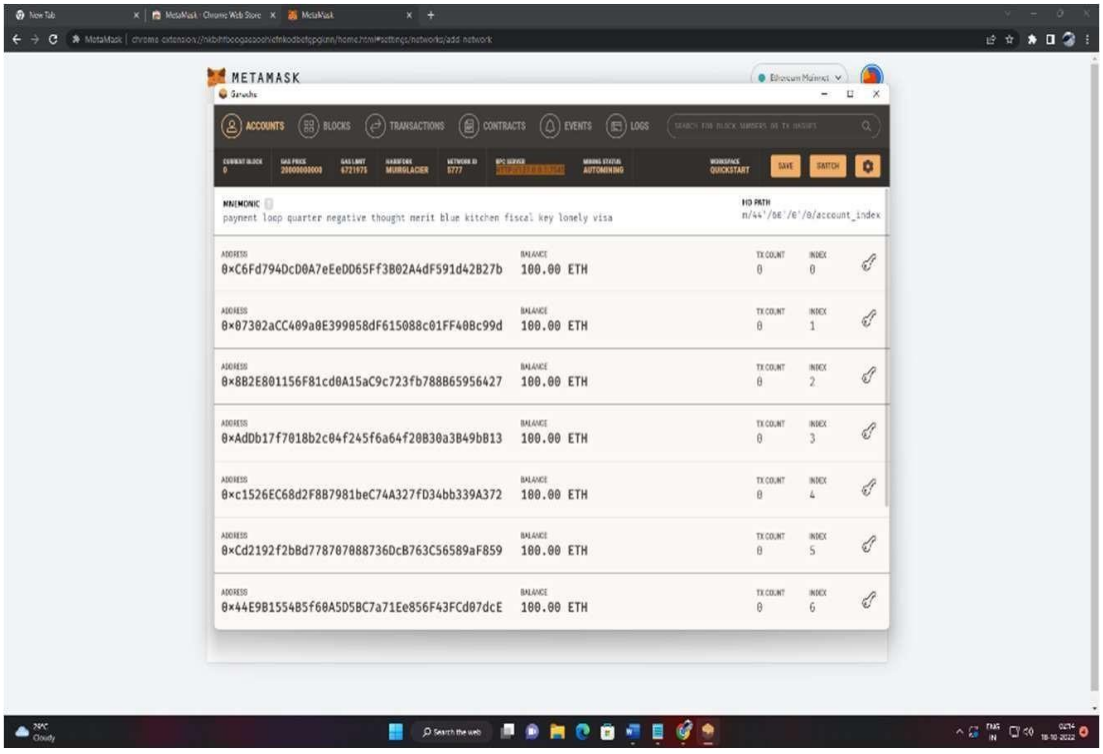


From here we will import ethers to our crypto wallet account.

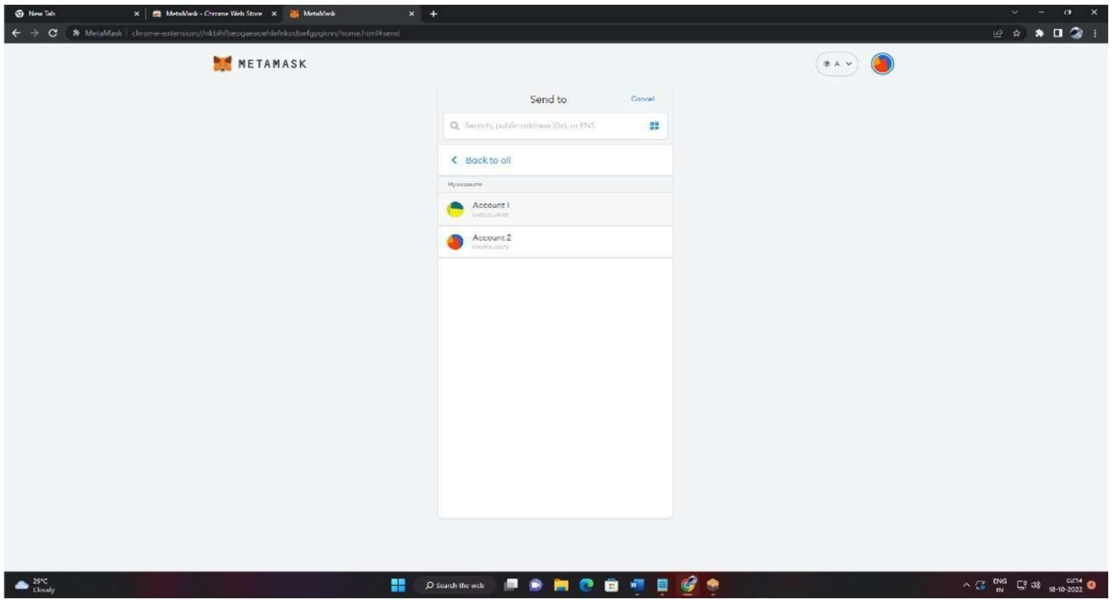


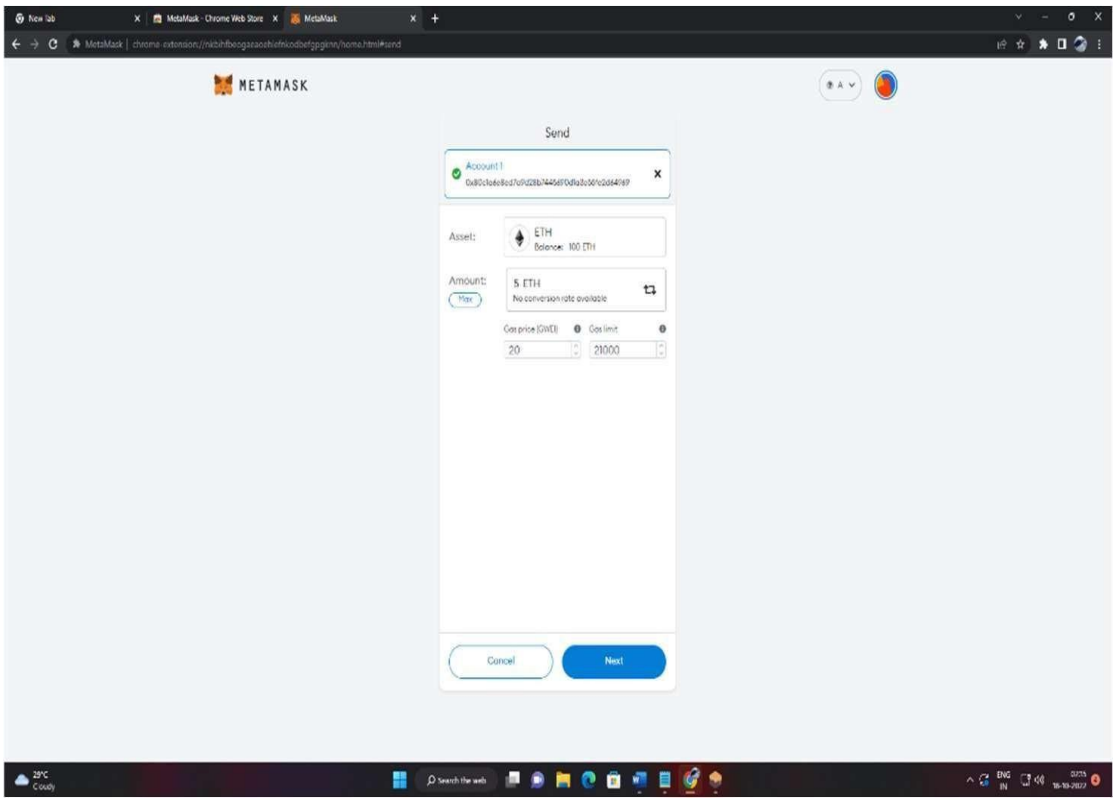
Here we will paste the private key for importing the account from the Ganache.

Now we'll perform the transaction between two different accounts using the metamask extension.



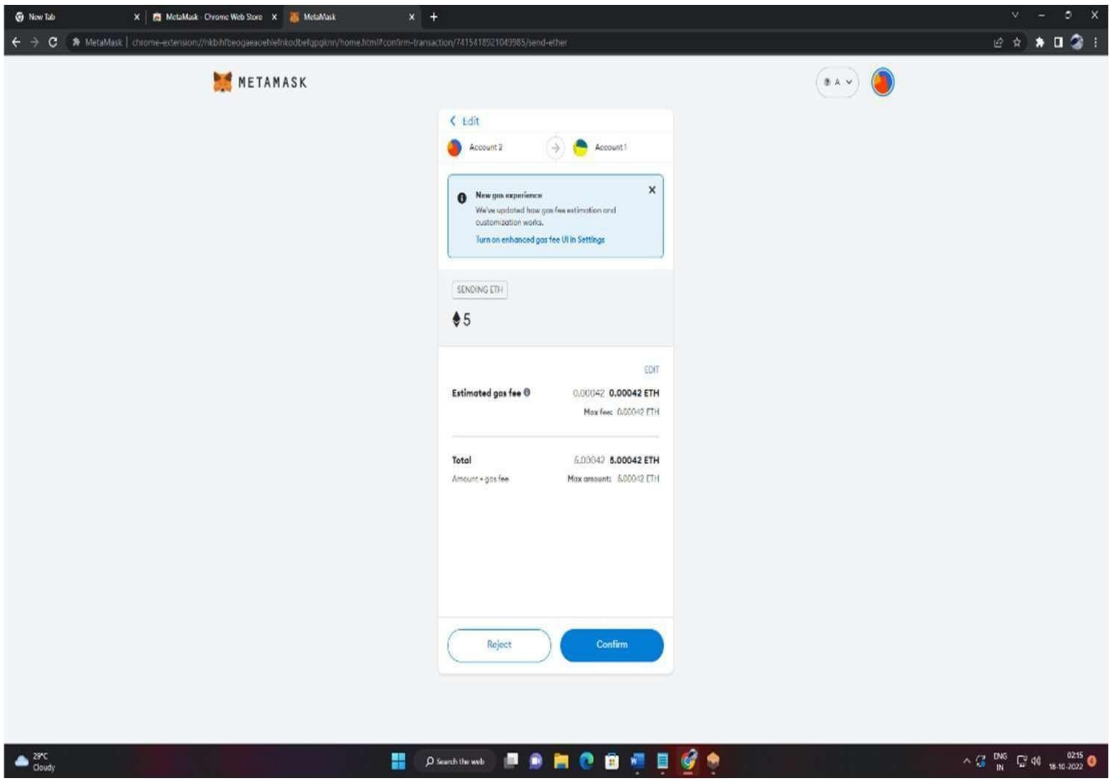
We'll perform the transaction between the first two accounts using the metamask as follows:



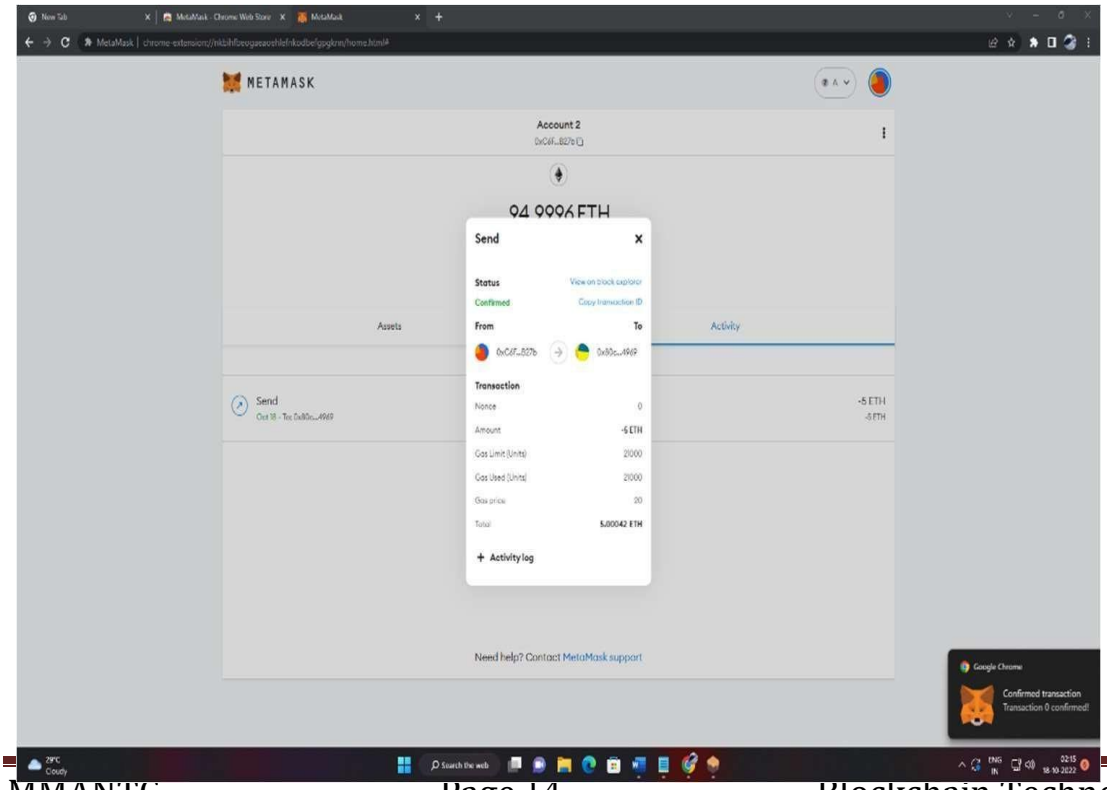


We are the transferring the ether from one account1 to the account 2, the above Picture

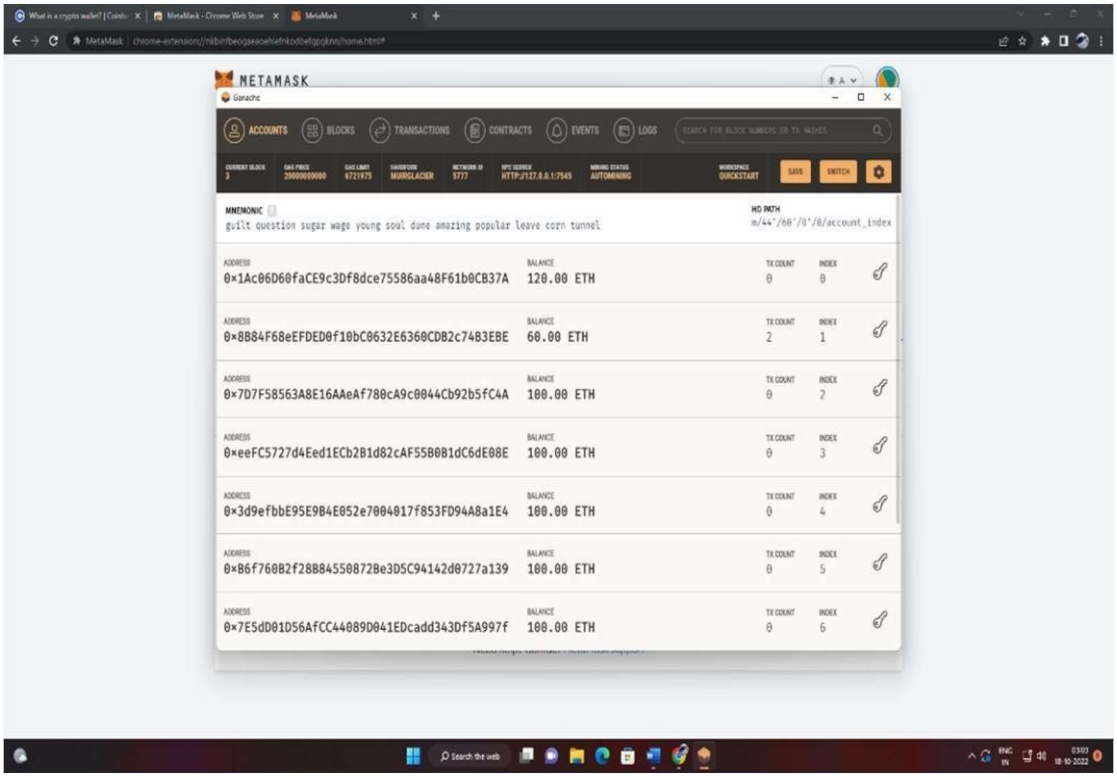
shows the details of the transaction.



The above picture shows the transaction details and the below picture shows the completion of the transaction and the details related to the transaction.



After the successful transaction the change in the wallet can be seen in the Ganache Platform.



Conclusion: We learn about importing crypto wallet into metamask and we learn about the transaction between the accounts.

ASSESMENT:

Exp. No. 02	**Roll No.**		**Date of starting**	
			Date of completion	
Remarks/ Grade				**Signature of subject teacher**

Practical No. 3

Title: Write a program in solidity to create Smart Contract.

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

Theory:

Solidity:

Solidity is a high-level language. The structure of smart contracts in solidity is very similar to the structure of classes in object-oriented languages. The solidity file has an extension.sol.

What are Smart Contracts?

Solidity's code is encapsulated in contracts which means a contract in Solidity is a collection of code (its functions) and data (its state) that resides at a specific address on the Ethereum blockchain. A contract is a fundamental block of building an application on Ethereum.

Example: In the below example, the aim is to deploy a Smart Contract for Marks Management System by using Solidity. In this contract, the details of every student like student ID, Name, Marks, etc. can be added and if one wants to give some bonus marks to students then they can also be added. After building the contract all the details of every student can be retrieved.

Approach:

a. The first step is to deploy the smart contract using the Remix IDE. After writing the code compile the code. When it is successfully compiled then deploy it. After deploying the contract, a deployed Contract is obtained and then add the student details one by one.
b. If bonus marks need to be added then add in the bonusMarks section after that click on stdCount and fetch the student details to call the stdRecords.
c. Add one or more new student details in this Smart Contract by the increment of stdCount.

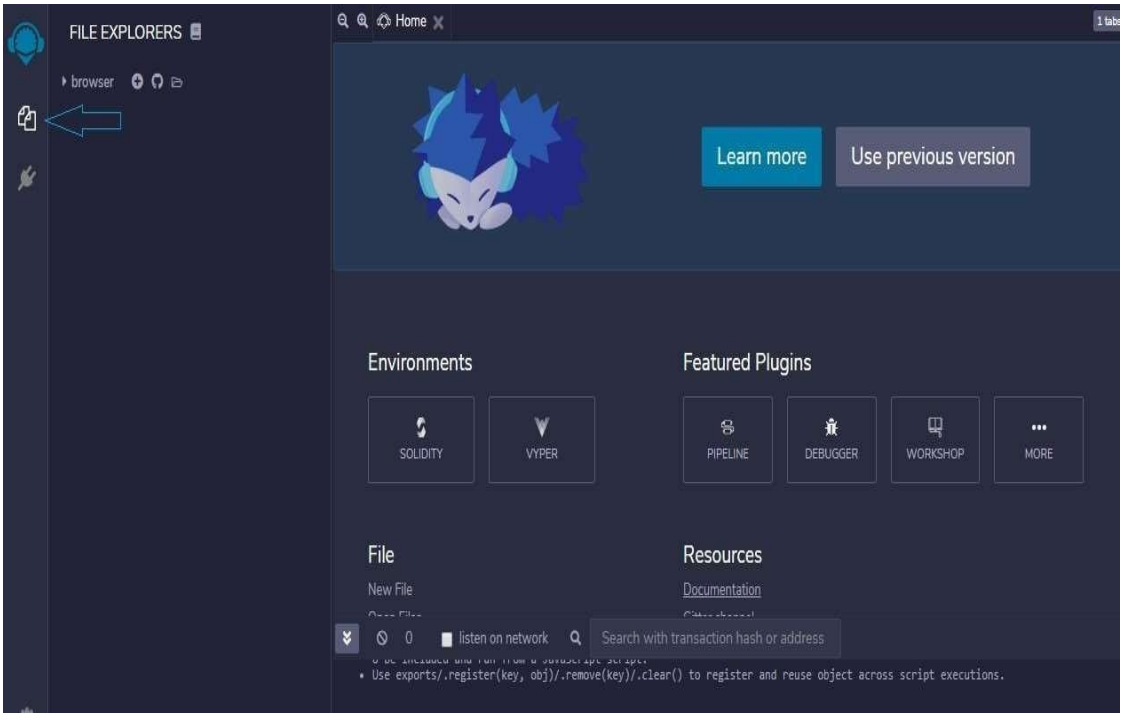
What is Remix-IDE?

The Remix is an Integrated Development Environment (IDE) for developing smart contracts in Solidity programming language. The IDE can be used to write, compile, and debug the Solidity code. It was written in JavaScript and supports testing, debugging and deploying smart contracts, and much more. Remix-IDE can be accessed in many different ways:

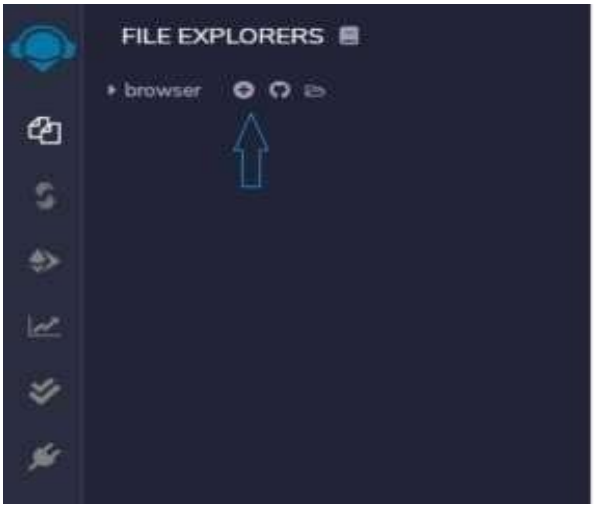
1) You can use it online in any browser of your choice, by entering the URL: https://remix.ethereum.org/ in the browser.
2) Or you can install it in your own system using this link.
3) The third way is to use Mist (the Ethereum Dapp browser).

Step:

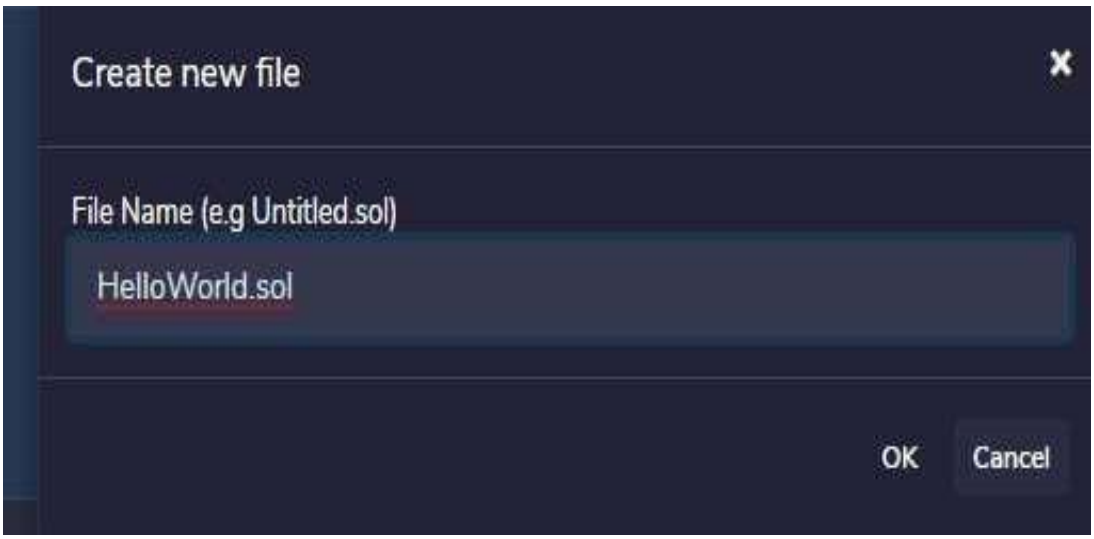
1. Open the remix-ide in your browser by typing: https://remix.ethereum.org/
2. You will be presented with following screen:



3. Click on the "file explorer" icon onto the left side bar (indicated by blue arrow in the above picture).
4. Select Solidity in the Environment and click + symbol right to the browser.



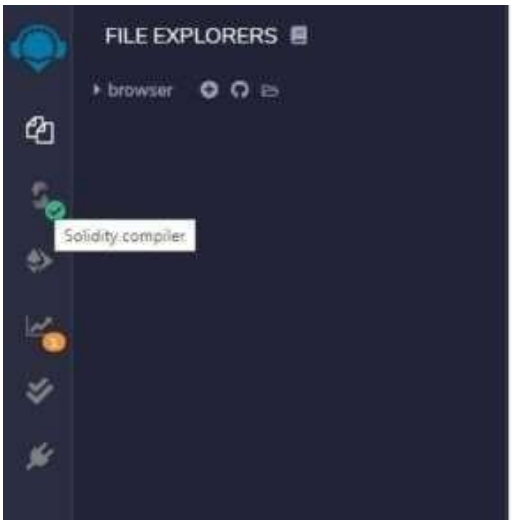
5. Type the file name “HelloWorld.sol” and enter the following code into it



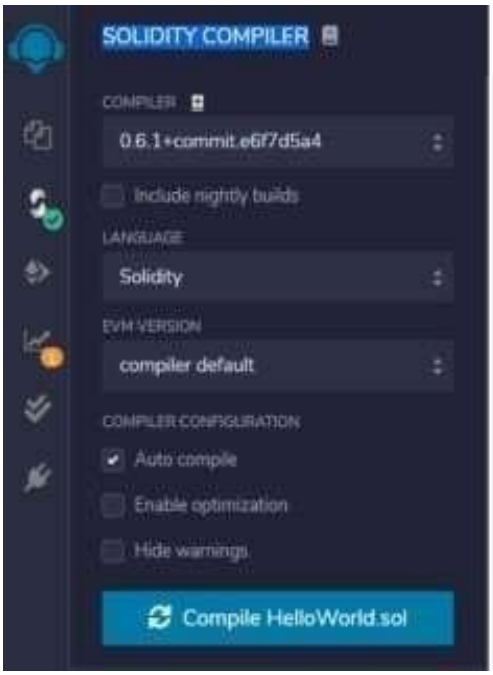
Code:

```
// My First Smart Contract pragma solidity >=0.5.0
<0.7.0;
contract HelloWorld {function get()public pure returns
(string memory){

return 'Hello Contracts';

   } }
```

6. Once done click the icon just below the “file explorer” icon as shown below:

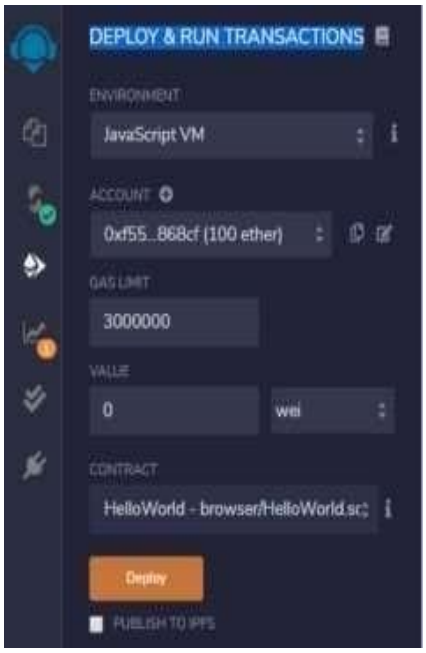


7. You will be presented with following screen:

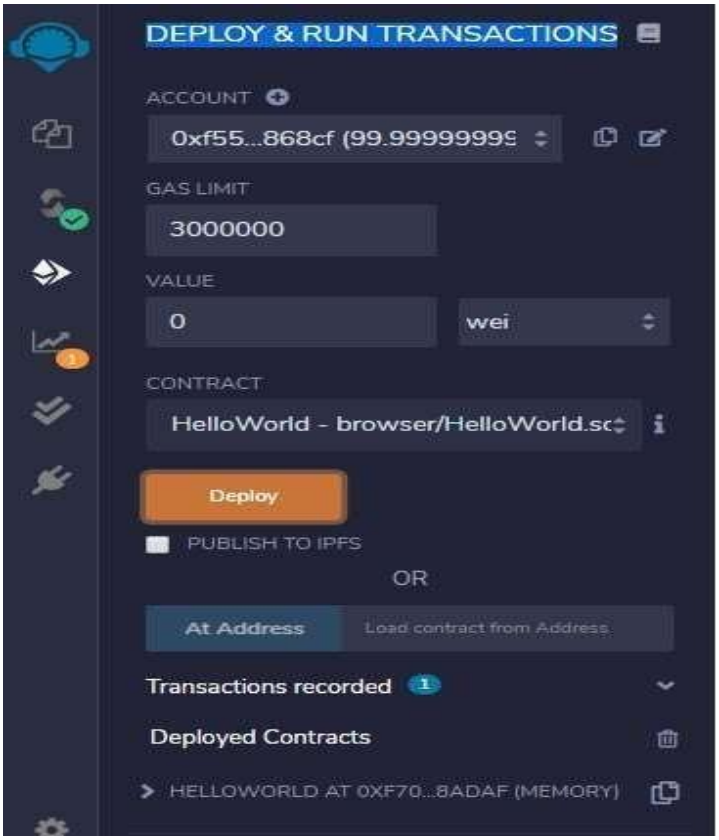


8. Click “Compile HelloWorld.sol”. Leave rest setting as it is.

9. Once compiled successfully click the icon below “Solidity Compiler” that is “Deploy and run transactions”. You will be welcomed by next screen:

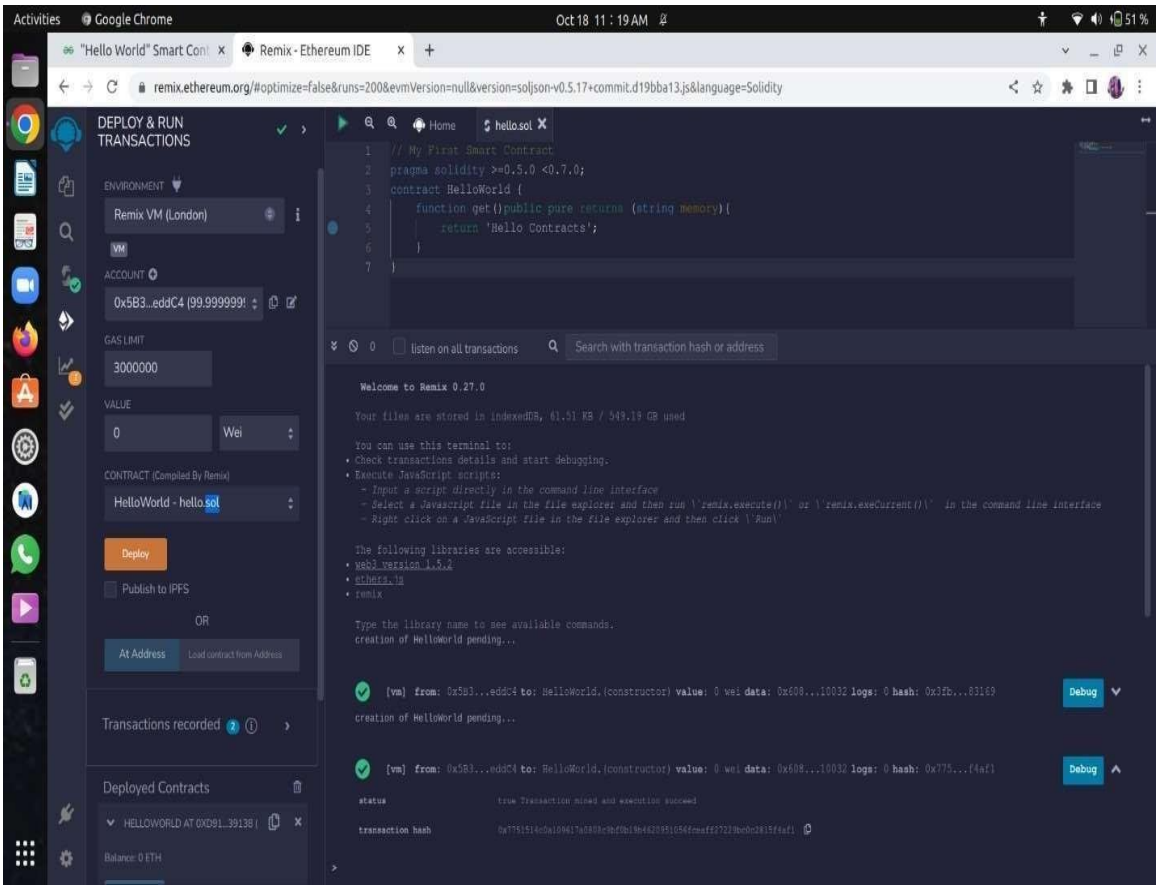


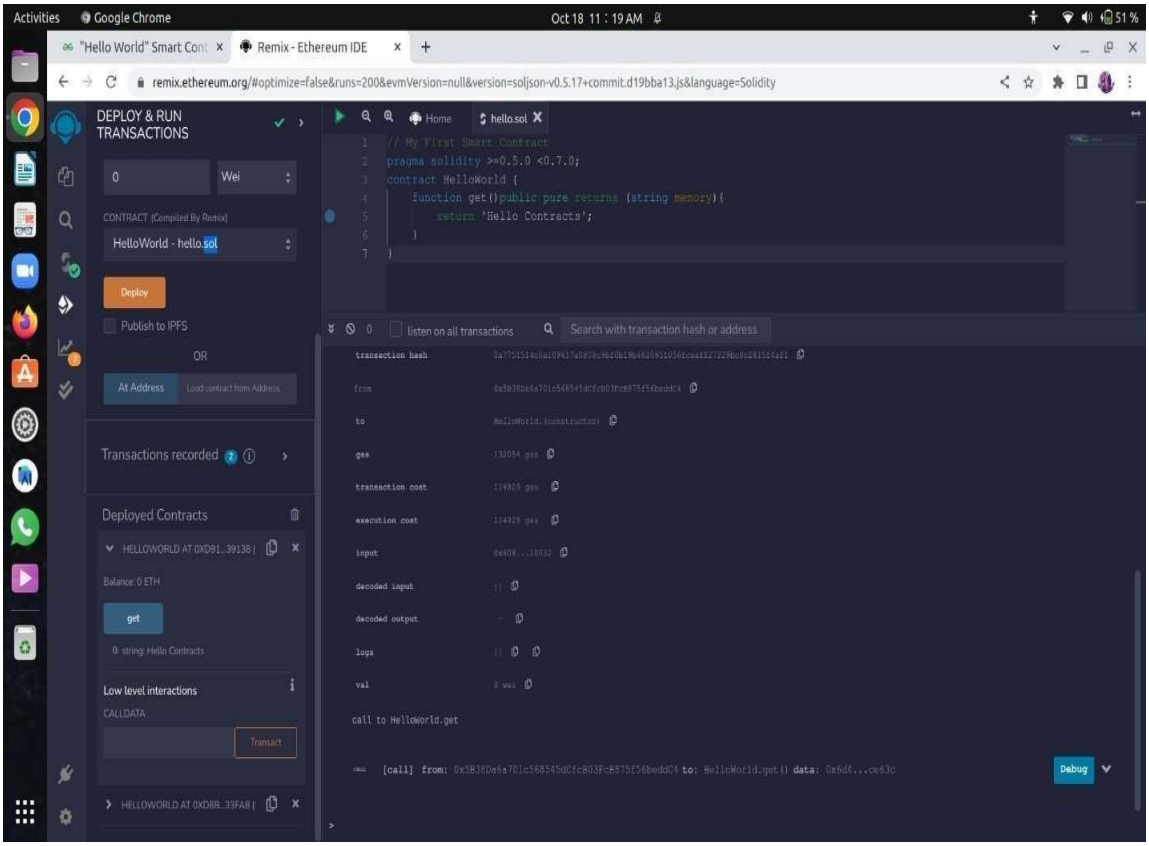
10. Without changing any of the values as shown above just click “Deploy” button to deploy your smart contract. Once deployed you will find your smart contract just below in “Deployed Contracts” heading:



11. Click ">" before your contract you will see a button "get" below as our contract has get function that returns a string. Click get button and you will get:

Output:





Conclusion: We have learned how to use data types like Structure, Array and fallback in solidity language.

ASSESMENT:

Exp. No. 03	**Roll No.**		**Date of starting**	
			Date of completion	
Remarks/ Grade				**Signature of subject teacher**

Practical No: 4

Title: Write a smart contract on a test network, for Bank account of a customer for following operations:

- Deposit money
- Withdraw Money
- Show balance

Theory:

What is smart contract?

A Smart Contract is a computer program that directly and automatically controls the transfer of digital assets between the parties under certain conditions. A smart contract works in the same way as a traditional contract while also automatically enforcing the contract. Smart contracts are programs that execute exactly as they are set up (coded, programmed) by their creators. Just like a traditional contract is enforceable by law, smart contracts are enforceable by code.

Up to this point, we have our smart contracts written and compiled without any errors. Let's see how we can deploy our smart contract on Rinke by test network. Deploying a contract on the main blockchain costs money (in the form of cryptocurrency), so we have test networks to develop and test our smart contracts. In the test network, fake ethers are added to our wallet to facilitate transactions. One such network provided by Ethereum is Rinke by.

What is a Test Network (test net)?

These are networks used by protocol developers or smart contract developers to test both protocol upgrades as well as potential smart contracts in a production-like environment before deployment to Main net.

There are multiple test nets available like Görli, Kovan, Rinke by and Ropsten. For this project we'll be using Rinke by.

Network Testing (or network performance testing), similar to Software Testing, is the process of analyzing and testing your network using a network performance test to identify bugs and performance issues, evaluate large network changes, and measure network performance.

Even the most robust networks experience network problems. That's why before and after every new service migration or deployment, every new application or network device, and honestly - just as a continuous practice, perform network testing using Network Monitoring to detect and troubleshoot problems as soon as they happen.

What is Remix-IDE?

The Remix is an Integrated Development Environment (IDE) for developing smart contracts in Solidity programming language. The IDE can be used to write, compile, and debug the Solidity code. It was written in JavaScript and supports testing, debugging and deploying smart contracts and much more

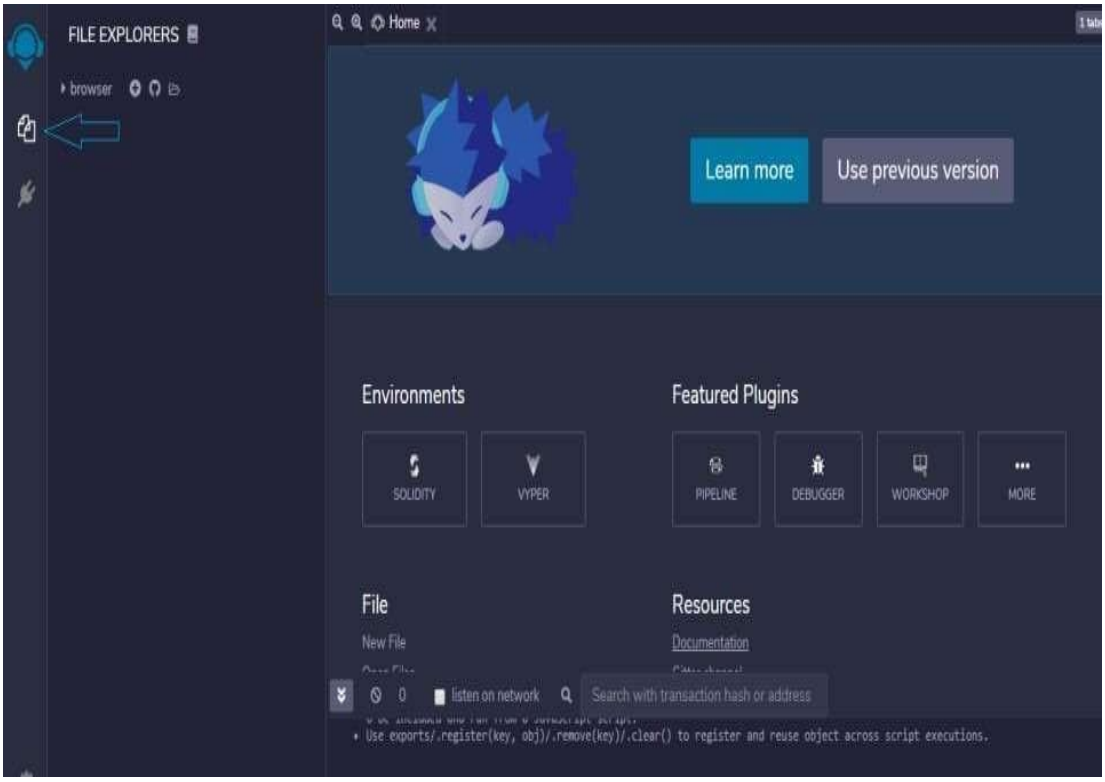
Remix-IDE can be accessed in many different ways:

1. You can use it online in any browser of your choice, by entering the URL: https://remix.ethereum.org/ in the browser.
2. Or you can install it in your own system using this link.
3. The third way is to use Mist (the Ethereum Dapp browser).

Step:

This article is all about writing your first smart contract in solidity using remix-ide which is a browser-based IDE.

1. Open the remix-ide in your browser by typing: https://remix.ethereum.org/
2. You will be presented with following screen:



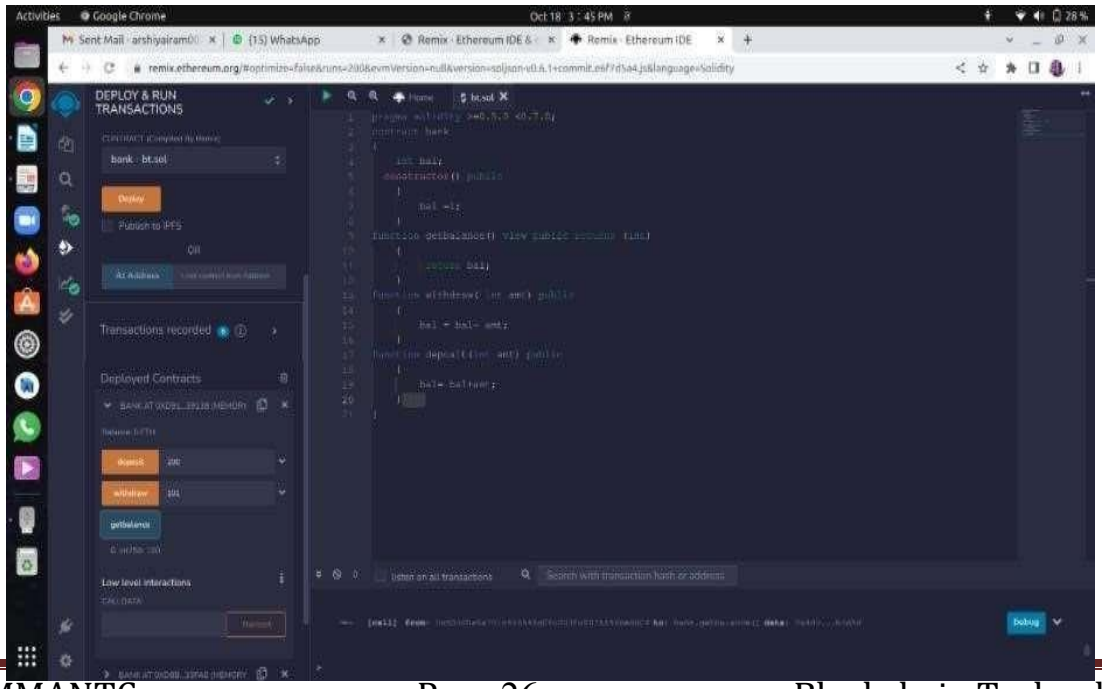
3. Click on the “file explorer” icon onto the left side bar (indicated by blue arrow in the above picture).
4. Select Solidity in the Environment and click + symbol right to the browser.

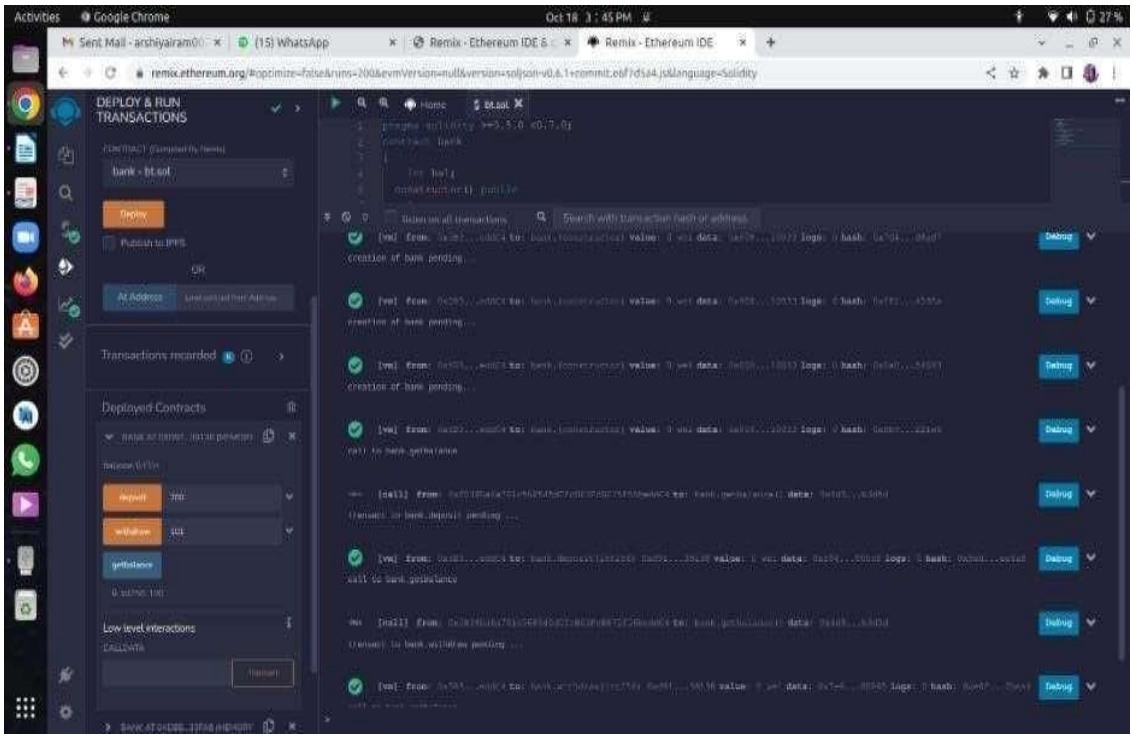
5. Type the file name and enter the following code into it.

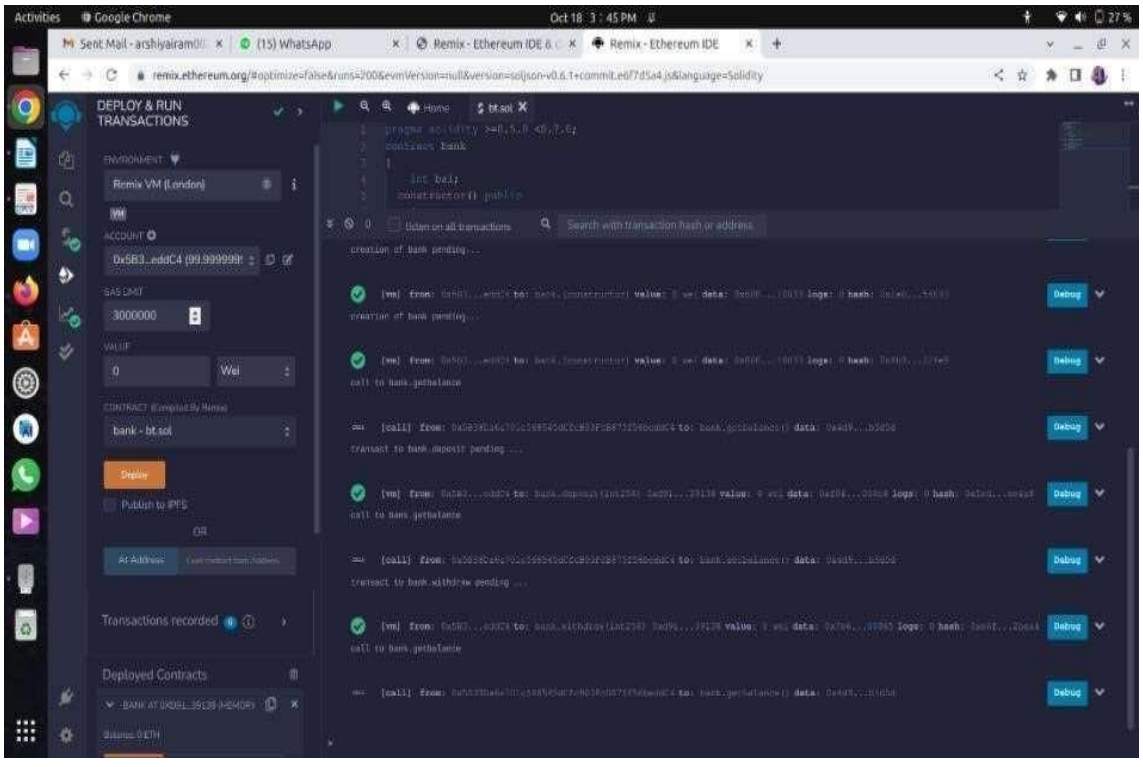
Code:

```
pragma solidity >=0.5.0 <0.7.0;
contract bank
{
   int bal;
 constructor() public
   {
bal =1;
   }
function getbalance() view public returns (int)
   {
       return bal;
   }
function withdraw( int amt) public
   {
      bal = bal- amt;
   }
function deposit(int amt) public
   {
      bal= bal+amt;
   }
}
```

Output:







Conclusion: We have learned how to write smart contract on remix ID using solidity language on a test network.

ASSESMENT:

Exp. No. 04	**Roll No.**		**Date of starting**	
			Date of completion	
Remarks/ Grade				**Signature of subject teacher**