Blockchain Sem VII

HBCC701 : Blockchain Development

Module - 4 : Public Blockchain (8 Hours)

Instructors : Dr. Nupur Giri & Mrs. Lifna C S

Topics to be covered

- Introduction to Public Blockchain,
- Ethereum and its Components -
  - Ethereum Virtual Machine (EVM),
  - Transaction,
  - Accounts,
- Architecture and Workflow,
- Mining in Ethereum,
- Comparison between Bitcoin and Ethereum
- Types of test-networks used in Ethereum,
- Transferring Ethers using Metamask, Mist Wallet,
- Ethereum frameworks,
- Case study of Ganache for Ethereum blockchain,
- Exploring etherscan.io and ether block structure

Introduction to Public Blockchain

- **does not have restrictions.**
- **Anyone with an internet connection can get access to the network** and **start validating blocks and sending transactions.**
- Offers incentive for users who validate the blocks.
- The network tends to use **Proof of Work or Proof of Stake consensus algorithms** for validating the transactions.
- It is a “Public” network in a true sense.
- It was the model that **Satoshi Nakamoto suggested back in 2009**.
- Later, enterprise companies, tweaked the nature of the decentralized ledger and introduced the private blockchains.
- Anyone can download the protocol anytime, without any permission from anyone.
- **it’s completely decentralized, no single organization controls the ecosystem.**
- Whereas, a private blockchain can be changed and altered by the owning organization.
- A public blockchain surpassed the necessity of a third party.
- A **self-governed, purely decentralized and autonomous digital public ledger.**

Courtesy : 101 Blockchain

Introduction to Public Blockchain

Characteristics:

- It is an **open network and permissionless**
  - where nodes can join and leave without the permission of anyone.
  - Every node **has access to read and write on the ledger**
  - Anyone can download and add nodes to the system
  - the ledger is shared and transparent.
- Includes a **protocol of incentive mechanism**
  - to ensure the correct operation of the blockchain system.
- It is **secure to the 51% rule.**
- It **offers anonymity**, which means no one can track your transactions back to you
- **No regulation** hence no limit to how one can use the platform for betterment.
- The technology is **fully decentralized in nature**
- **Everyone can change current business models** through the reduction in the use of middlemen.
- It is **not necessary to maintain servers or system administrators**.
- Hence there is **considerable cost reduction for the businesses**.

Courtesy : 101 Blockchain

Introduction to Public Blockchain

Advantages:

- **Transparency**:
  - Public blockchains are transparent and open for everyone to access.
  - This makes the ledger accessible to all,
  - eliminating chances of corruption and ensuring transparency.
- **Security:**
  - Public blockchains are designed to operate with maximum security.
  - The decentralized nature of the network makes it difficult for hackers to compromise the system.
- **Empowerment:**
  - Allows all participants to validate transactions without any central authority overlooking their actions.
- **Immutability:**
  - meaning no one can tamper with the system, ensuring that transactions are secure.

Courtesy : Geeks4Geeks

Introduction to Public Blockchain

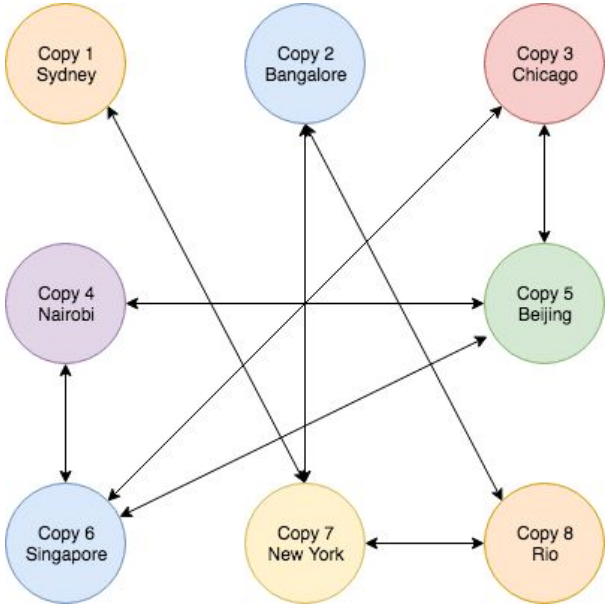
Disadvantages:

- **Power Consumption:**
  - require a lot of computational power due to their decentralized nature.
  - This increases energy consumption and can be detrimental to the environment.
- **Scalability:**
  - With more users on the blockchain, the network becomes burdened with more transactions, leading to scalability issues.
- **Conspiracy:**
  - Due to decentralized nature, no one knows who validates the transactions,
  - increasing the risk of potential conspiracy.
- **Transactions:**
  - Can be slow due to the time it takes to process all transactions on the network.
- **Acceptance:**
  - Due to the openness and transparency, it can be difficult for governments to accept them as they are not controlled by authorities

Courtesy : Geeks4Geeks

What is Ethereum?

- a **public, blockchain based distributed computing platform.**
- as **one big computer** made up of small computers around the world.
- Eg: One can write applications and run them on this global computer.
- The platform **guarantees that your application will always run without any downtime, censorship, fraud or third-party interference.**
- Ethereum blockchain can **also transfer money between 2 parties without a central authority**



What is Ethereum?

- A Blockchain network that introduced a built-in **Turing-complete programming language** that can be **used for creating** various decentralized applications(also called **Dapps**).
- Ethereum network is fueled by its own **cryptocurrency called 'ether'.**
- Allow the implementation of **smart contracts.** Smart contracts can be thought of as 'cryptographic bank lockers' which contain certain values.
- These cryptographic lockers can only be unlocked when certain conditions are met.
- Ethereum is a network that can be applied to various other sectors.
- called **Blockchain 2.0** → it proved that blockchain can be used beyond the financial sector.
- The consensus mechanism used in Ethereum is **Proof of Stakes(PoS)**, which is **more energy efficient** than, Proof of Work(PoW).
- PoS depends on the **amount of stake a node holds**.

History of Ethereum

2013

- Ethereum was **first described in Vitalik Buterin's white paper**
- Goal of **developing decentralized applications.**

2014:

- **EVM was specified in a paper by Gavin Wood**, and the formal development of the software also began.

2015:

- Ethereum **created its genesis block** marking the **official launch of the platform.**

2018:

- Ethereum **took second place** in Bitcoin **in terms of market capitalization.**

2021:

- a major network upgrade named **London included Ethereum improvement proposal 1559**
- **introduced a mechanism** for **reducing transaction fee volatility.**

2022:

- Ethereum has **shifted from PoW(Proof-of-Work) to PoS(Proof-of-State)**
- Also known as **Ethereum Merge**.
- It has **reduced Ethereum's energy consumption by ~ 99.95%.**

Features of Ethereum

1. Smart contracts:
   a. Ethereum allows the creation and deployment of smart contracts.
   b. Smart contracts are created mainly using a programming language called **solidity.**
2. Ethereum Virtual Machine (EVM):
   a. **a runtime environment for compiling and deploying Ethereum-based smart contracts.**
3. Ether:
   a. **cryptocurrency of the Ethereum network**.
   b. **only acceptable form of payment for transaction fees** on the Ethereum network.
4. Decentralized applications (Daaps):
   a. **has its backend code running on a decentralized peer-to-peer network**.
   b. Has a frontend and UI to make calls and query data from its backend.
   c. They **operate on Ethereum** and **perform the same function irrespective of the environment** in which they get executed.
5. Decentralized autonomous organizations (DAOs):
   a. **works in a democratic and decentralized** fashion.
   b. **relies on smart contracts for decision-making** within the organization.

Real-World Applications of Ethereum

1. **Voting**:
   a. Voting systems are **adopting Ethereum.**
   b. The **results of polls are available publicly, ensuring a transparent fair system thus eliminating voting malpractices.**
2. **Agreements**:
   a. With Ethereum smart contracts, agreements and contracts **can be maintained and executed without any alteration.**
   b. Ethereum can be **used for creating smart contracts and for digitally recording transactions based on them.**
3. **Banking systems**:
   a. **Due to the decentralized nature** of the Ethereum blockchain it becomes **challenging for hackers to gain unauthorized access to the network**.
   b. **Makes payments on the Ethereum network secure**
4. **Shipping**:
   a. **Ethereum provides a tracking framework** that helps with the **tracking of cargo and prevents goods from being misplaced.**
5. **Crowdfunding**:
   a. **helps to increase trust and information symmetry**.
   b. It **creates many possibilities for startups by raising funds** to create their own digital cryptocurrency.

Benefits of Ethereum

1. **Availability:**
   - As the Ethereum network is decentralized so **there is no downtime**.
   - Even if one node goes down other computing nodes are available.
2. **Privacy:** Users don't need to enter their personal credentials while using the network for exchanges, thus **allowing them to remain anonymous**.
3. **Security**: Ethereum is **designed to be unhackable,** as the hackers have to get control of the majority of the network nodes to exploit the network.
4. **Less ambiguity**: The **smart contracts that are used as a basis for trade and agreement** on Ethereum ensure stronger contracts than traditional contracts which require follow-through and interpretation.
5. **Rapid deployment:** On Ethereum decentralized networks, enterprises can easily deploy and manage private blockchain networks instead of coding blockchain implementation from scratch.
6. **Network size**: Ethereum network **can work with hundreds of nodes and millions of users.**
7. **Data coordination:** Ethereum **decentralized architecture better allocates information** so that the **network participants don't have to rely on a central entity** to manage the system and mediate transactions.

Drawbacks of Ethereum

- **Complicated programming language:** Learning solidity from programming smart contracts on Ethereum can be challenging and one of the main concerns is the scarcity of beginner-friendly classes.
- **Volatile cryptocurrency:** Ethereum investing can be risky as the price of Ether is very volatile, resulting in significant gains as well as a significant loss.
- **Low transaction rate**:
  a. **Bitcoin** has an average transaction rate of **7TPS**
  b. **Ethereum** has an average speed of **15 TPS** which is almost double that of bitcoin but it is still not enough.

Components of Ethereum Network

1. Ethereum Node
2. Ethereum Client
3. Ether
4. Gas
5. Ethereum Accounts
6. Nonce
7. Storage Root
8. Ethash
9. Transactions
10. Ethereum Virtual Machine (EVM)

Components of Ethereum Network - (1) Ethereum Node

- a **computer that is running the software client**.
- Nodes **communicate with one another in order to validate transactions and record data** about the status of the blockchain.
- **Responsible for storing, validating, and trading data.**
- Each node keeps its **own copy of the blockchain** and **strives to verify** that it matches the copies of all the other nodes.
- Every node on the network **must process any action that requires a new block to be added** to the blockchain.
- This network of continually communicating nodes **allows us to avoid relying on a single source of truth and all of the challenges it entails.**
- A new block is **added based on whether or not the majority of nodes accept it.**

Components of Ethereum Network - (1) Ethereum Node (Types)

1. **Full Node**:
   - verify and validate each and every transaction that takes place inside the network
   - **maintain the state / a full copy** of the blockchain.
   - When a smart contract transaction occurs,
     - Full nodes also **execute all of the instructions in the smart contract**.
     - It determines whether or not the smart contract execution is **producing the expected results.**
   - It **keeps receiving copies of the entire blockchain** including its transactions which are stored locally and keeps the latest state of transaction with itself.
   - Eg: **Person A performs a transaction to person B**,
     - transaction is added to the blockchain,
     - Full nodes verify whether the transaction complies with all the Ethereum specifications,
     - Maintain the latest state of the blockchain by storing or removing the specification if it does not comply.
   - **Example of a discarder transaction** is when a person transfers X ETH to another person but their account contains less ETH.

Components of Ethereum Network - (1) Ethereum Node (Types)

2. **Archive Nodes:**
   - **complete nodes that have the "archive mode" option enabled.**
   - While a **Full Node only stores the latest state of the transaction,**
   - the Archive nodes **hold all of the blockchain's history data dating back to the genesis block**.
   - **used when blocks prior to the latest 128 blocks are required.**
   - Eg : using functions like **eth_getBalance** of a historic address would require an archive node, as will interacting with smart contracts launched far earlier in the blockchain.
   - Archive Nodes **memory requirement** :
     - **require more than 6 Terabytes of space**, contrary to Full node which only requires a little over 500 Gigabytes of disk space.
     - **are not useful for average people**,
     - they are **effective in the application of block exploring, wallet vending, and chain analytics**.

Components of Ethereum Network - (1) Ethereum Node (Types)

3. **Light Nodes:**

- does not hold the complete current blockchain state
- stores only the block header.
- suitable for low memory and computational devices since maintaining a light node involve the least investment in hardware, running costs, and technical skill.
- Light nodes **rely on full nodes to function.**
- These nodes **do not need to run continuously or read and publish a large amount of data on the blockchain.**
- It provides **an easy way to create a wallet, especially for beginners.**
- Eg : **Solid-State Drives (SSD)** cannot afford to store the gigabytes of data that other nodes take.
- But there are some limitations of light nodes which cannot be denied, there is no guarantee that the light wallet provider will be online when it is needed.

Components of Ethereum Network - (2) Ethereum Client

- **software program** that is used **to implement the Ethereum specification**
- **connect itself with other Ethereum clients** over a peer-to-peer network.
- Different Ethereum clients can communicate with one another if they follow the reference specification and the defined communication protocols.
- These interactions among different clients in the network **take place using various programming languages**
  - like Geth (Go), OpenEthereum (Rust),
  - Nethermind (C#, .NET), Besu (Java),
  - Erigon (Go/Multi).
- The **yellow paper** is the Ethereum protocol that allows anybody to run a client to construct a node.
- **Ethereum sets standard behaviors that all Ethereum clients must adhere to**
- **Ethereum's specs** enabled the blockchain to allow for different, but interoperable, software implementations of an Ethereum client by providing standard documentation and simple language.

Components of Ethereum Network - (2) Ethereum Client (Types)

1. **Full Client:**
   - **save the complete Ethereum blockchain**,
   - which **might take several days to synchronize** and
   - takes a **massive amount of disc space** – more than 1 Terabyte, according to the most recent estimates.
   - **Enable connected nodes to conduct all network functions**, including as
     - mining,
     - transaction
     - block-header validation,
     - smart contract execution.

Components of Ethereum Network - (2) Ethereum Client (Types)

2. **Light Client:**
   - **do not always need to necessary keep all of the data,**
   - when data storage and performance are concerns, developers utilize the “light clients”.
   - Light clients **provide a portion of full client capability.**
   - they can provide quick delivery and free up data storage space.
   - The functionality of a light client is adapted to the purposes of the Ethereum client.
   - **widely used within wallets to maintain private keys and Ethereum addresses.**
   - **manage smart contract interactions and transaction broadcasts**.
   - **useful for web3 instances** within JavaScript objects, Dapp browsers
   - **obtaining the exchange rate data.**

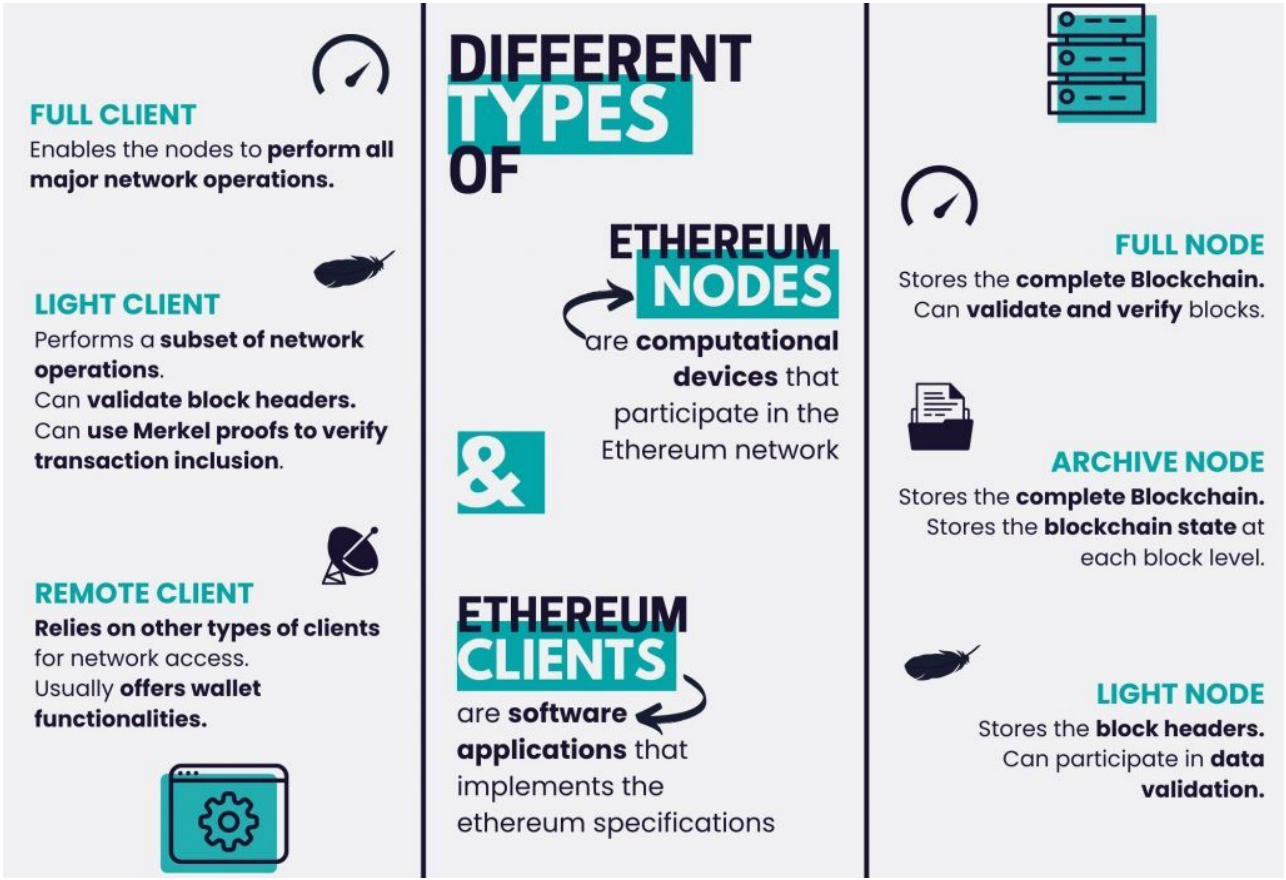
Components of Ethereum Network - (2) Ethereum Client (Types)

3. **Remote Client:**
   - A remote client is **much like a light client**.
   - The primary distinction is that a **remote client does not keep its own copy of the blockchain or validate transactions or block headers.**
   - Remote clients, on the other hand, rely entirely on a full or light client to have access to the Ethereum blockchain network.
   - These clients are **mostly used as wallets for transmitting and receiving transactions.**

Components of Ethereum Network - Node Vs Client

Ethereum Node	Ethereum Client
A machine running Ethereum client software is referred to as an “Ethereum Node”	A client is an Ethereum implementation that validates all transactions in each block, ensuring the network’s security and data accuracy
The three types of Ethereum Nodes are Full, Light, Archive, and Miner Nodes.	The three types of Ethereum Clients are Full, Light, and Remote Clients
The Ethereum node operating system allows us to access the internet	The Ethereum client computer allows a user to access the node operating system

Components of Ethereum Network - Node Vs Client



Courtesy : Chainstack

Components of Ethereum Network -

	Pros	Cons
Light nodes	• Portable • Resource-efficient • User-friendly	• Do not validate the network • Do not propagate blocks • Do not maintain consensus • Less secure
Full nodes	• Validate the network • Propagate blocks • Maintain consensus • More secure	• Resource-heavy • Harder to maintain • Less user-friendly
Pruned	Flexible storage	Need to revalidate old blocks
Archive	Carry full history	Resource and storage heavy
Mining	• Easily trackable involvement • Can pool with others to increase reward rate	• High and wasteful energy consumption • High equipment cost and barrier to entry
Staking	• Low barrier to entry • Low energy consumption	• Reward system based on luck • Low transparency in staking pools
Masternodes	• Balanced network benefits and rewards • Lower maintenance costs	• High initial investment • Difficult setup process

Courtesy : Chainstack

Components of Ethereum Network - (3) Ether

- type of **cryptocurrency used in the Ethereum network**
- It is a **peer-to-peer currency**
- It **tracks and promotes each transaction** in the network.
- It is the **second-largest cryptocurrency in the world.**
- Other cryptocurrencies can be used to get ether tokens, but vice versa is not true.
    - It means that **ether tokens can't be interchanged by other cryptocurrencies** to render computing power for Ethereum transactions.
- paid as a commission for any execution that affects the state in Ethereum.
- **used** in the Ethereum algorithm **as an incentive for miners to blocks to the blockchain using PoW**
- only currency that can be **used to pay transaction costs**, which go to miners as well.
- Aside from paying for transactions, **ether is often used to purchase gas**, which is used to pay for the computation of any transaction on the Ethereum network.

Components of Ethereum Network - (4) Gas

- **An internal currency** of the Ethereum network.
- We need gas to run applications on the Ethereum network, much as we need gas to run a vehicle.
- To complete every transaction on the Ethereum network, a consumer must first make a payment—send out ethers—and **the intermediate monetary value is known as gas.**
- Gas is a unit of measurement on the Ethereum network **for the computing power used to execute a smart contract or a transaction.**
- The price of gas is **very low compared to Ether.**
- **The execution and resource utilization costs are predetermined in Ethereum** in terms of Gas units, called **gwei.**

Components of Ethereum Network - Ether Denominations

Value (in wei)	Exponent	Common Name	SI Name
1	1	wei	wei
1,000	10^3	babbage	kilowei or femtoether
1,000,000	10^6	lovelace	megawei or picoether
1,000,000,000	10^9	shannon	gigawei or nanoether
1,000,000,000,000	10^{12}	szabo	microether or micro
1,000,000,000,000,000	10^{15}	finney	milliether or milli
1,000,000,000,000,000,000	*10^{18}*	*ether*	*ether*
1,000,000,000,000,000,000,000	10^{21}	grand	kiloether
1,000,000,000,000,000,000,000,000	10^{24}		megaether

Courtesy : Zastrin

Components of Ethereum Network - (5) Ethereum Accounts

- similar to a bank account,
- but for ethers or ETH, where Ethereum can be held, transferred to other accounts,
- can also be **used to execute smart contracts**.
- An entity that is composed of **an Ethereum address along with a private key.**
  - The first 20 bytes of the SHA3 hashed public key is the Ethereum address.
- Ethereum has two types of accounts:
  - **Externally owned account (EOA):**
  - **Contract Account**:

Components of Ethereum Network - (5) Ethereum Accounts (Types)

Ethereum has two types of accounts:

1. **Externally owned account (EOA):**
   - **controlled by private keys**.
   - Each EOA **has a public-private key pair**.
   - The **users can send messages by creating and signing transactions**.

Advantages of EOA

1. Transactions from an external account to a contract account **can trigger code that can execute many different actions,** such as transferring tokens or even creating a new contract.
2. Externally Owned Accounts cannot list incoming transactions.

Components of Ethereum Network - (5) Ethereum Accounts (Types)

2. **Contract Account**:
   - **controlled by contract codes**.
   - These **codes are stored with the account**.
   - Each contract account **has an ether balance associated with it**.
   - The contract code of these accounts **gets activated every time a transaction from an EOA or a message from another contract is received by it**.
   - When the contract code activates, **it allows to read/write the message to the local storage, send messages and create contracts.**
   - **Types of Contract Accounts :**
     i. Simple Account: The account is *created and owned by a single account holder*.
     ii. Multisig (multisignature) Account: A Multisig Wallet *contains several owner Accounts*, one of which is also the creator Account.

Components of Ethereum Network - (5) Ethereum Accounts (Types)

Advantages of CA:

1. A contract account can list incoming transactions.
2. Contract accounts can be set up as Multisig Accounts.
3. A Multisig Account can be structured such that it has a daily limit that you specify, and only if the daily limit is exceeded will multiple signatures be required.

Disadvantages of CA:

1. Creating contract accounts costs gas because they use the valuable computational and storage resource of the network.
2. Contract accounts can't initiate new transactions on their own. Instead, contract accounts can only fire transactions in response to other transactions they have received either from an externally owned account or from another contract account.

Components of Ethereum Network - (5) Ethereum Account (EOA Vs CA)

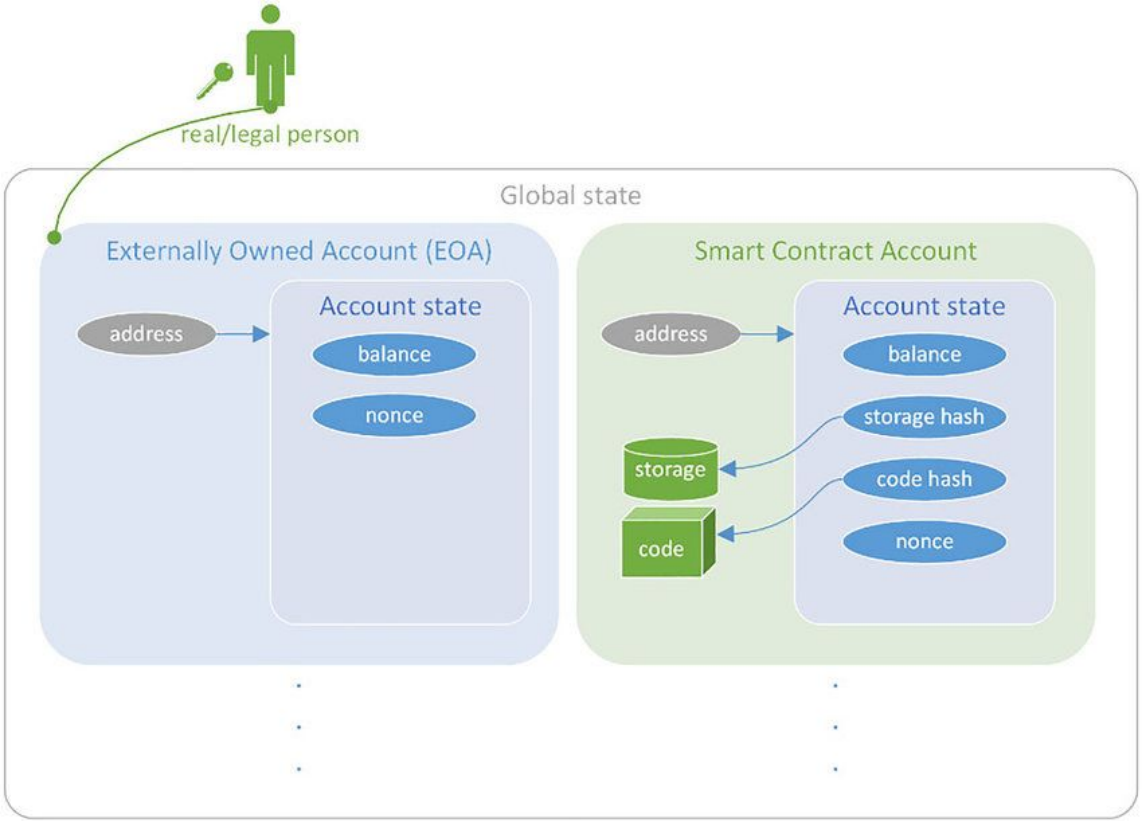
Externally Owned Accounts	Contract Account
Controlled by third party	Controlled by Contract Code
Private Key is needed to access EOAs	No key is needed to access Contract Accounts
EOAs are created automatically on creating wallet	CA require EOAs to be activated
EOA doesn’t have a code associated with it	CA have their own associated code
No execution fee is associated with EOAs	Execution fee is associated with CAs
Code hash = empty string	Code hash represents the code associated with tha account

Components of Ethereum Network - (5) Ethereum Account (Field)

Different Fields in Ethereum Accounts

1. **Nonce:**
   - The nonce in an Ethereum account indicated the number of transactions that have been sent from that account.
   - This ensures that each transaction is made only once by taking count every time it takes place.
2. **Ether Balance:** the amount of ether present in an ether repository of the current ether account.
3. **Contract Code**:
   - This is non-mandatory to fill, in case it is present since not all accounts have a contract code.
   - But note, that they cannot be altered once executed.
4. **Storage:** This field remains not filled unless mentioned.
5. **Code Hash**:
   - hash that refers to the code present in that Ethereum account
   - since no code is associated with externally owned Ethereum accounts, ⇒ **code hash = empty string.**

Components of Ethereum Network - (5) Ethereum Account (Field)



Courtesy : ResearchGate

Components of Ethereum Network - (5) Ethereum Accounts & Key Pairs

An Ethereum account is a **private-public key pair** that may be **linked to a blockchain address**.

Private Key

- It is a “owned” or “externally owned” account if the **private key is known and controlled by someone**.
- **Contract accounts** do not have a private key connected with them, although EOAs have.
- Control and access to one’s assets and contracts are granted through the EOA private key.
- The **user keeps the private key safe.**

Public Key

- Account’s **public key is open**.
- This key serves as the account’s identity.
- A one-way cryptographic function is **used to produce the public key from the private key.**

For example, if you create an account on Ethereum,

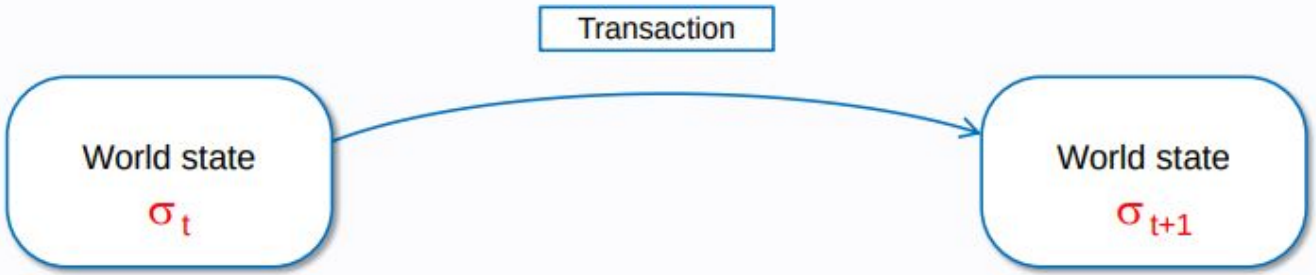
- **Retain the private key**
- **Share the public key.** As transactions between accounts are completed using public keys.

Components of Ethereum Network - Nonce, Storage Root, Ethash

6. **Nonce**
   - **For EOAs,** nonce means the number of transactions via this account.
   - **For CA,** nonce means the number of contracts generated via this account.
7. **Storage Root**
   - It is the **main root node of a Merkle tree**.
   - **Hash of all details of the account** is stored here.
   - The **root of the Merkle tree** is the verification of all transactions.
8. **Ethash**
   - **PoW algorithm for Ethereum 1.0**
   - most **recent version of Dagger-Hashimoto**

Components of Ethereum Network - Transactions

A transaction-based state machine



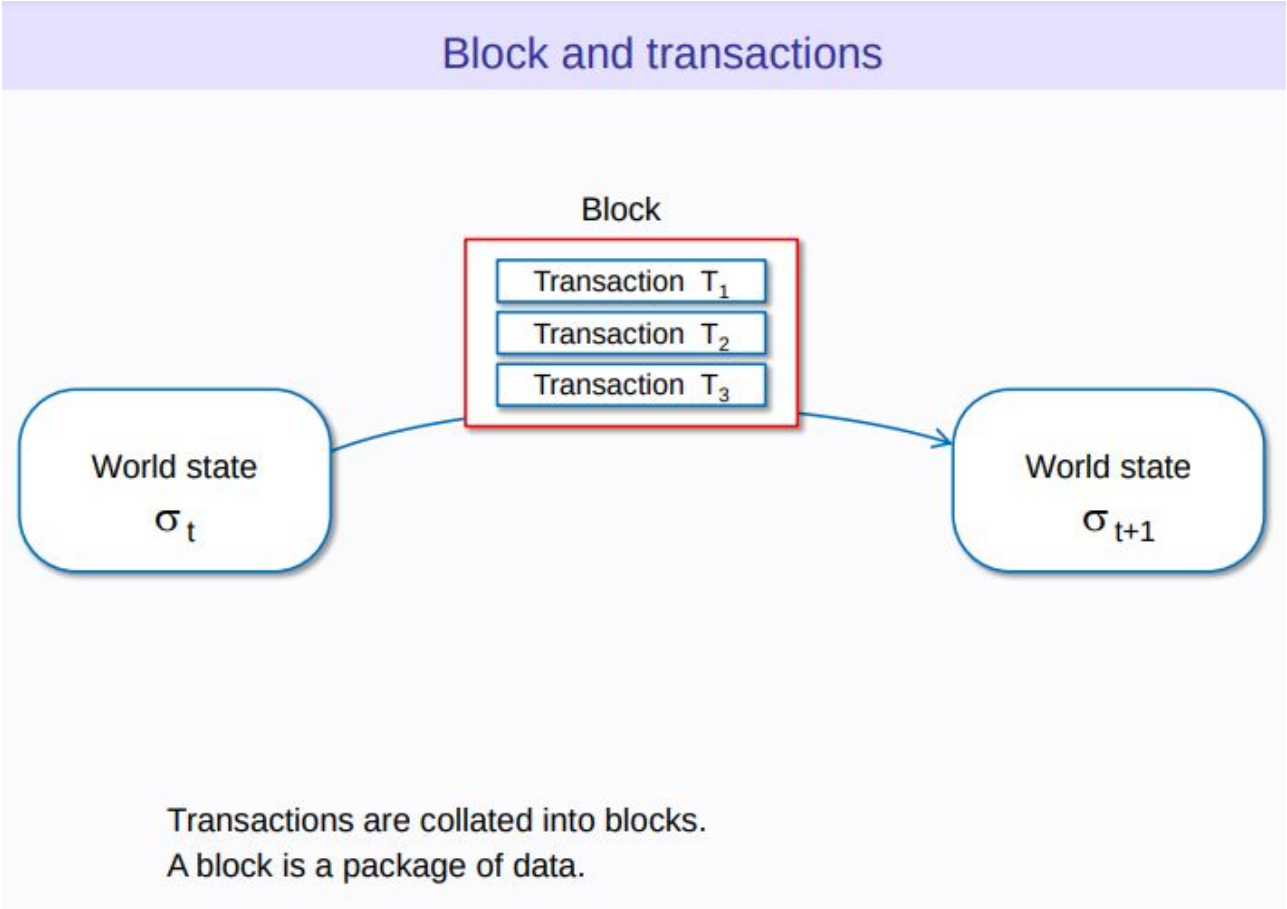
Ethereum can be viewed as a transaction-based state machine.

Components of Ethereum Network - Transactions

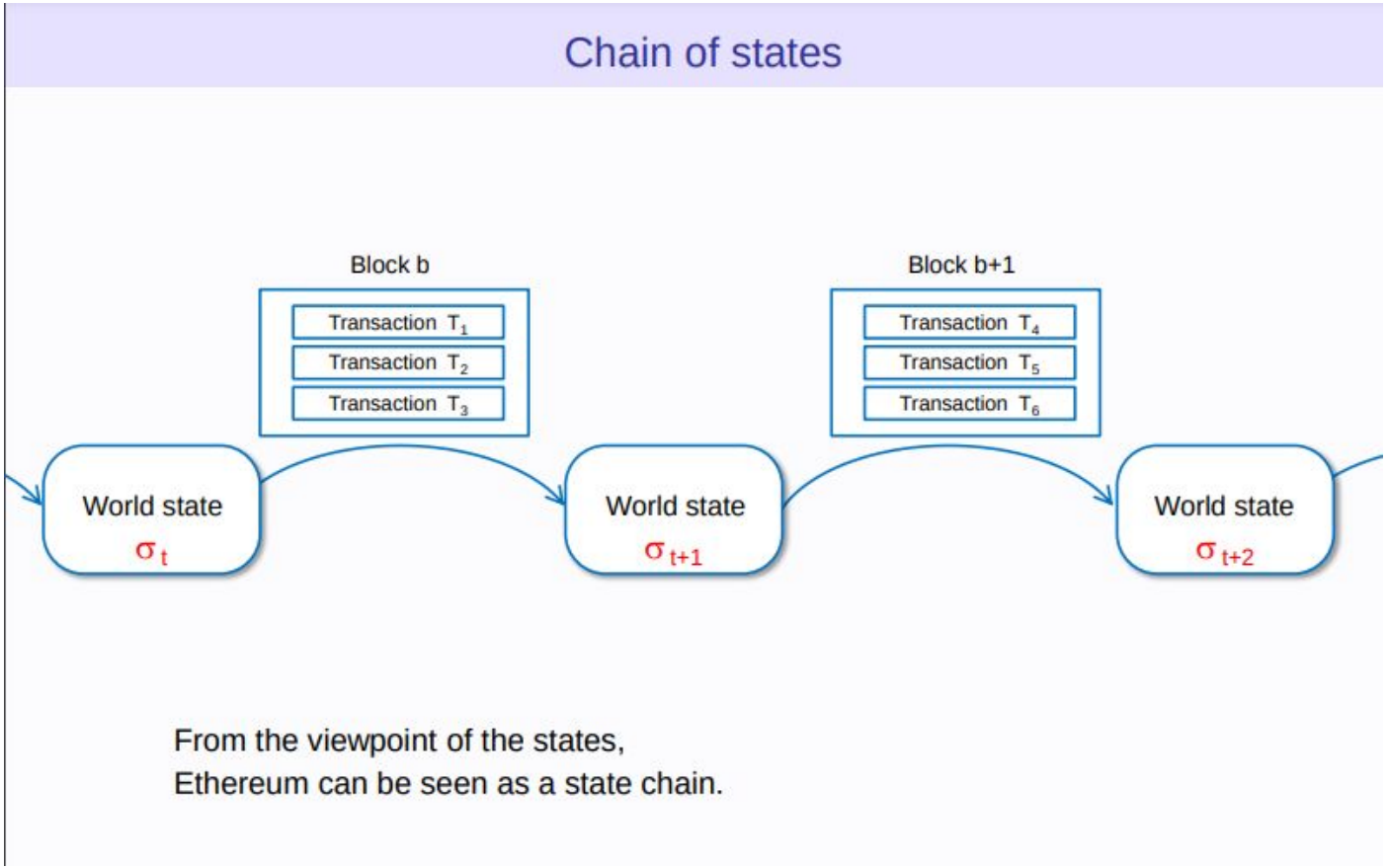
A transaction-based state machine

A transaction represents a valid arc between two states.

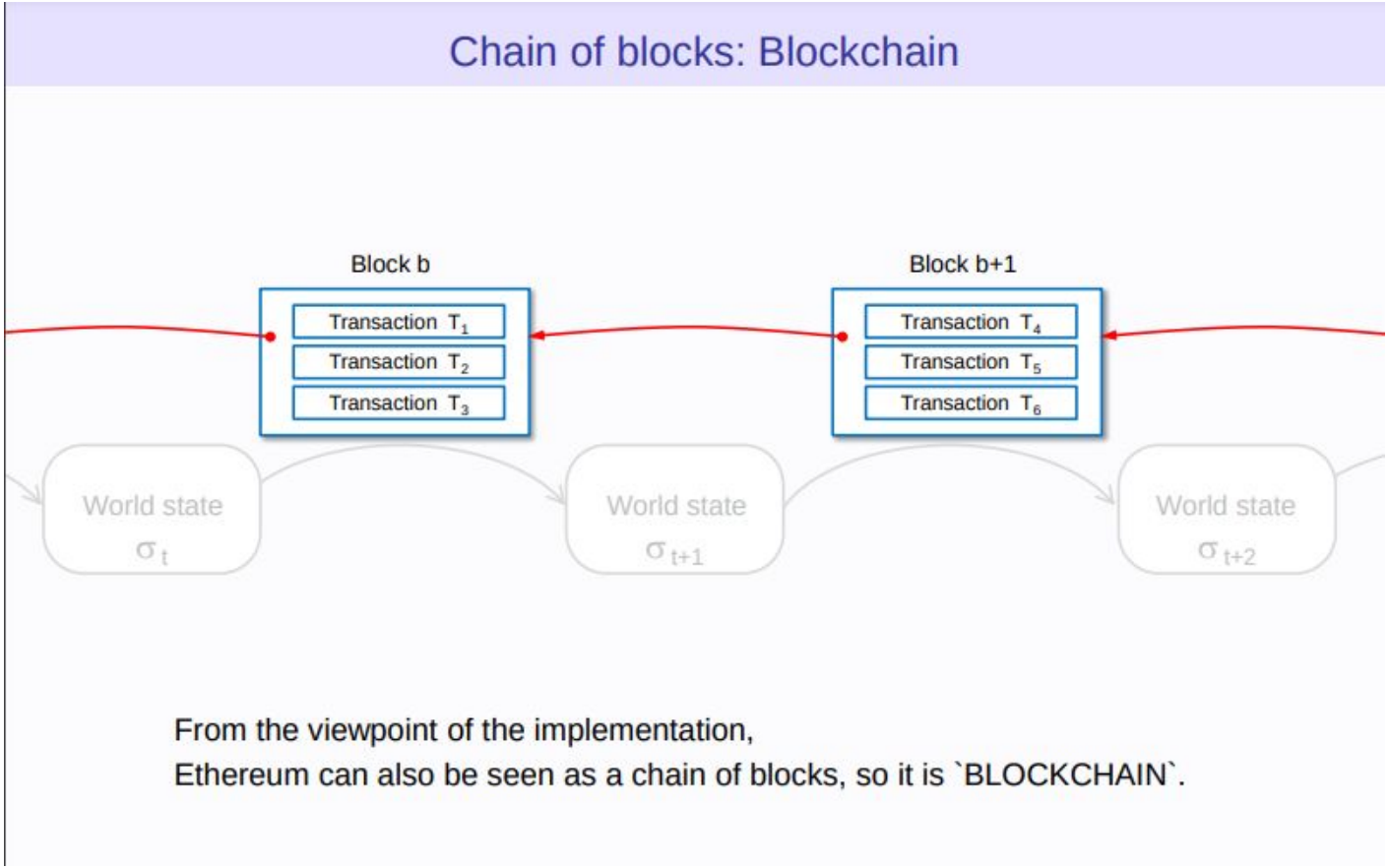
Components of Ethereum Network - Transactions



Components of Ethereum Network - Transactions

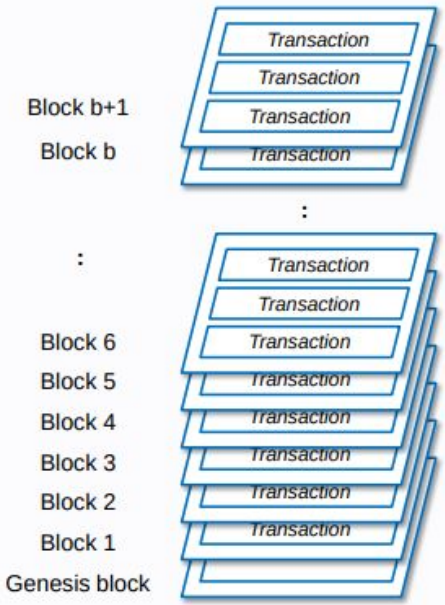


Components of Ethereum Network - Transactions



Components of Ethereum Network - Transactions

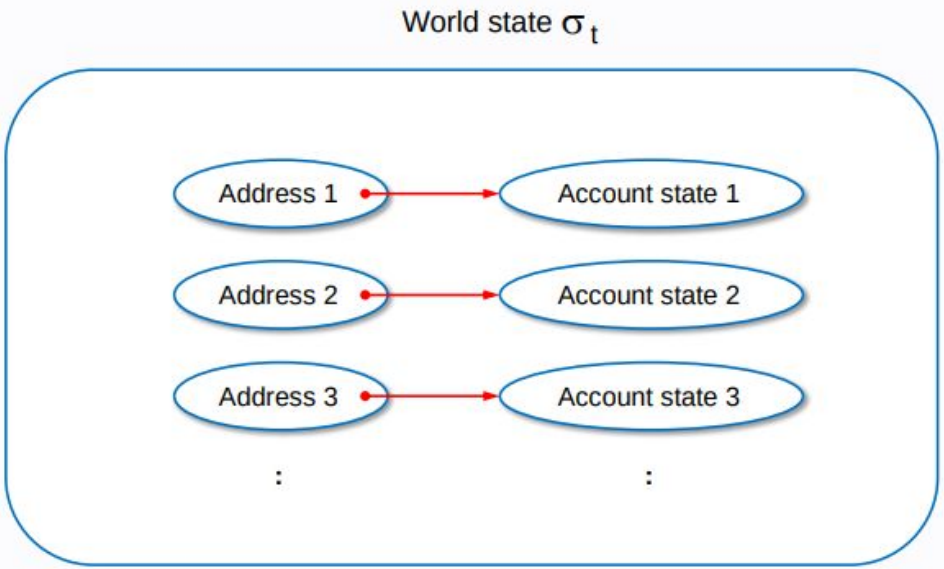
Stack of transactions : Ledger



From the viewpoint of the ledger,
Ethereum can also be seen as a stack of transactions.

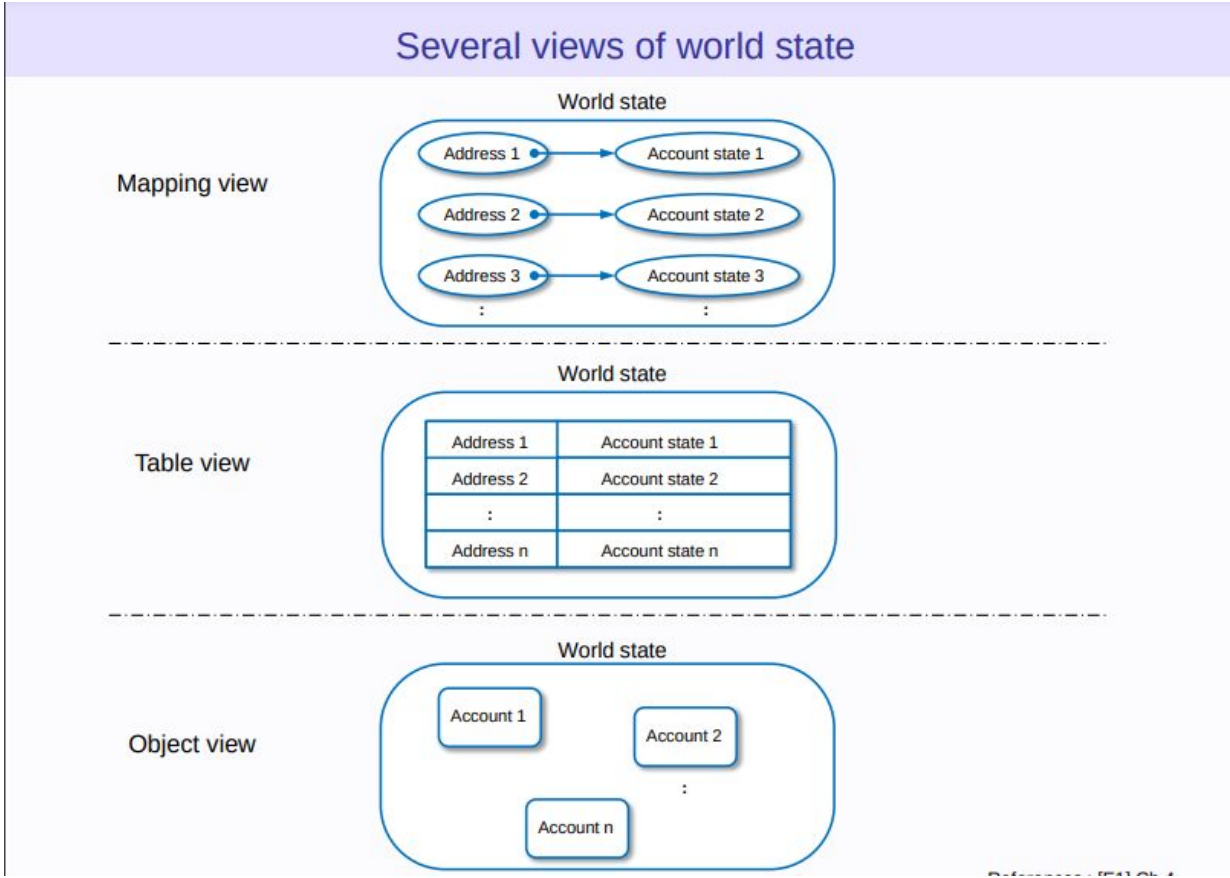
Components of Ethereum Network - Transactions

World state

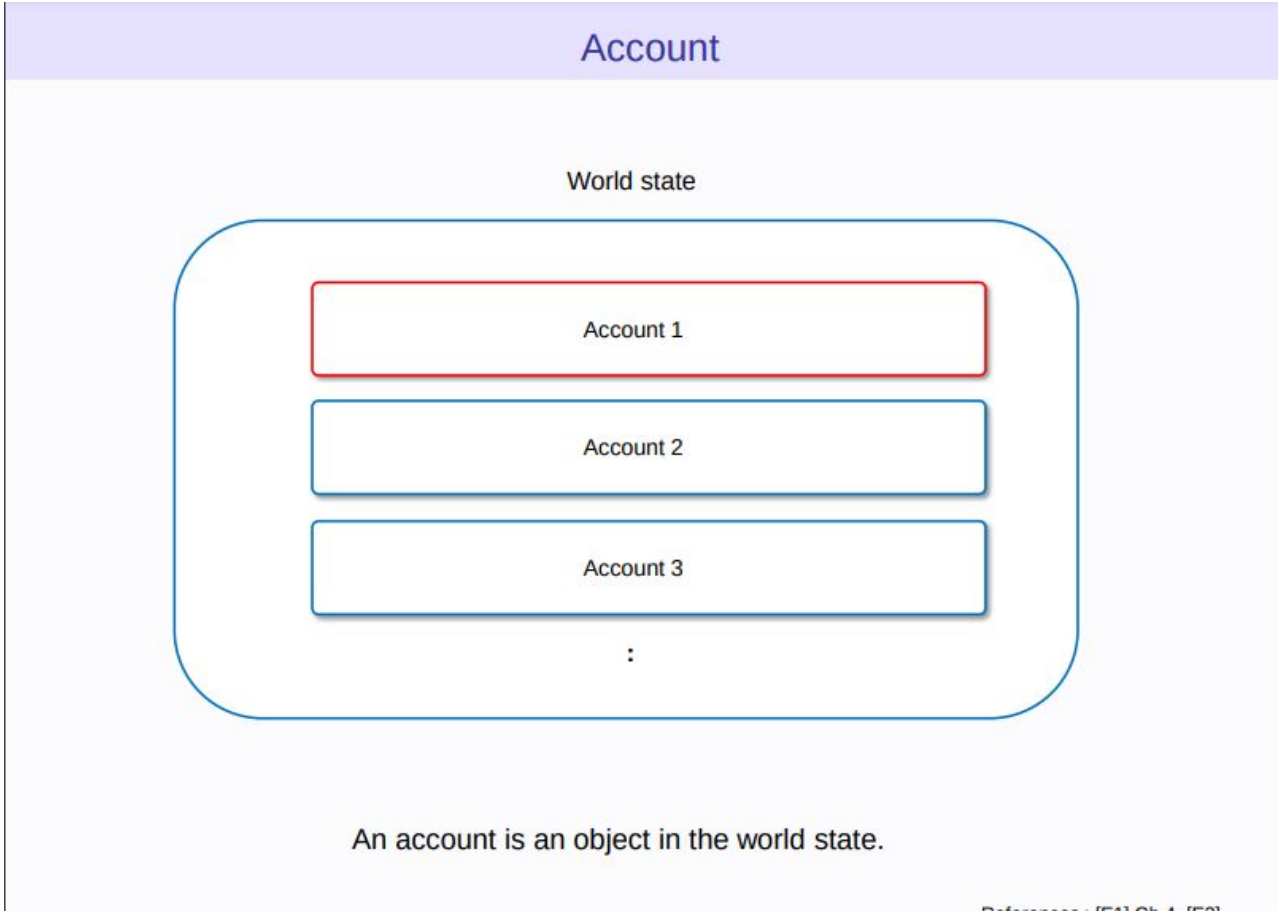


The world state is a mapping between address and account state.

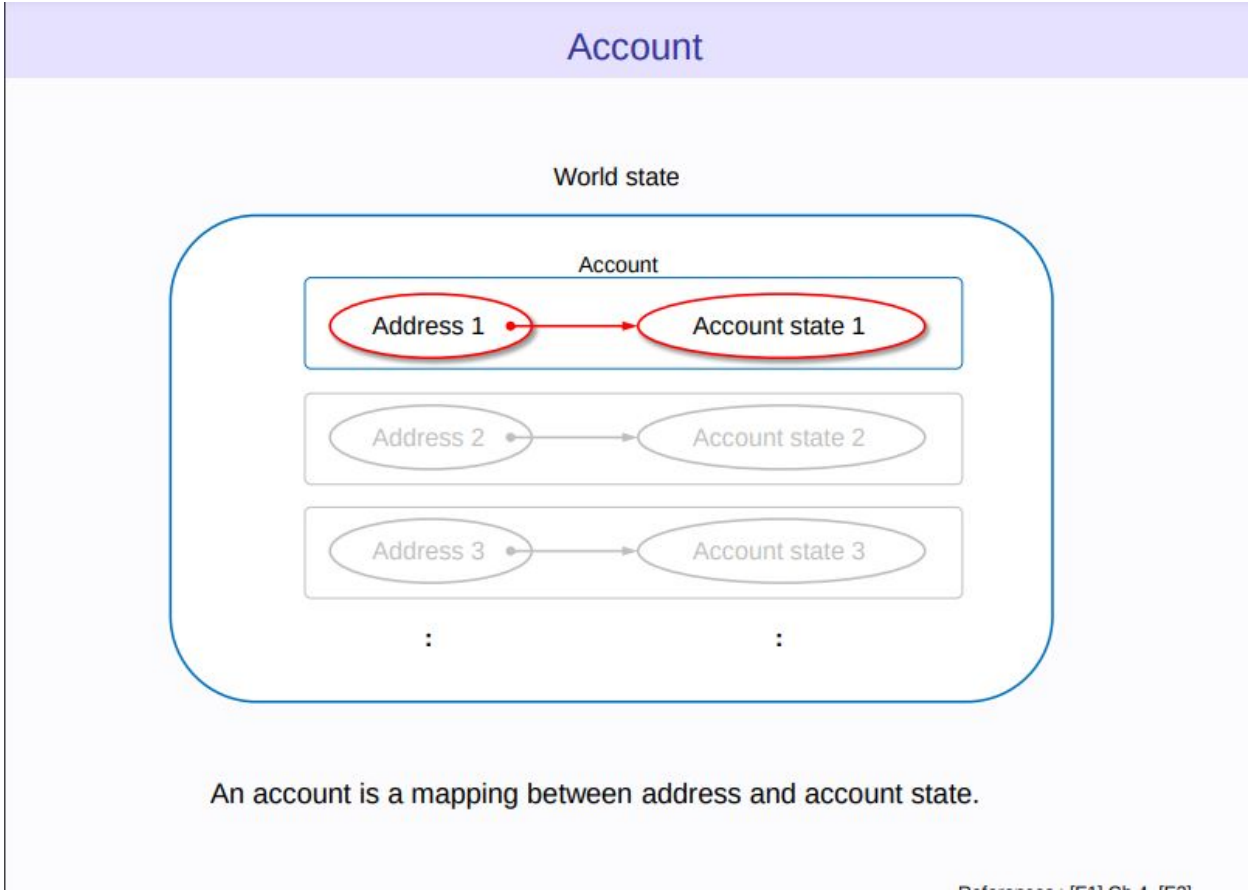
Components of Ethereum Network - Transactions



Components of Ethereum Network - Transactions

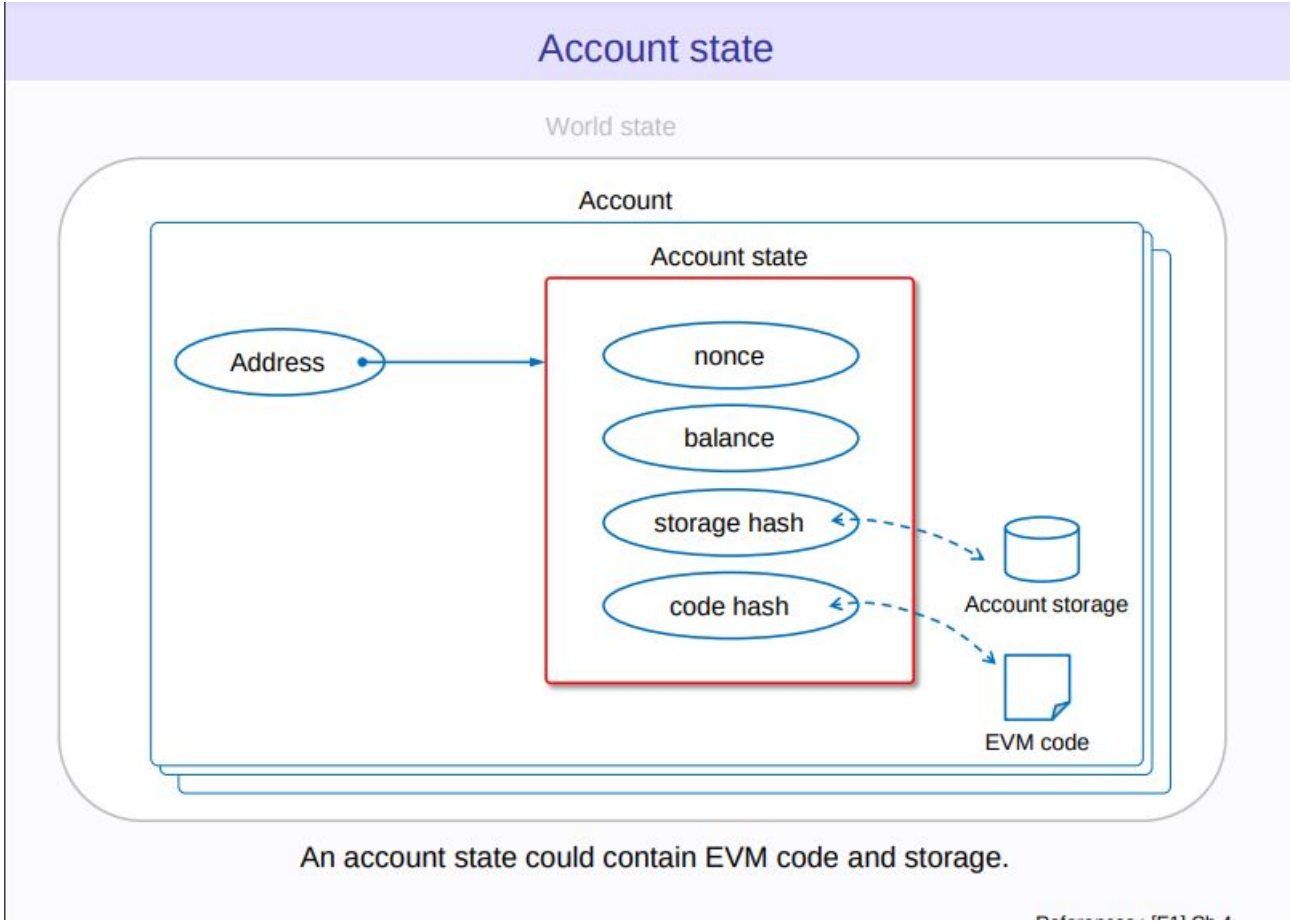


Components of Ethereum Network - Transactions



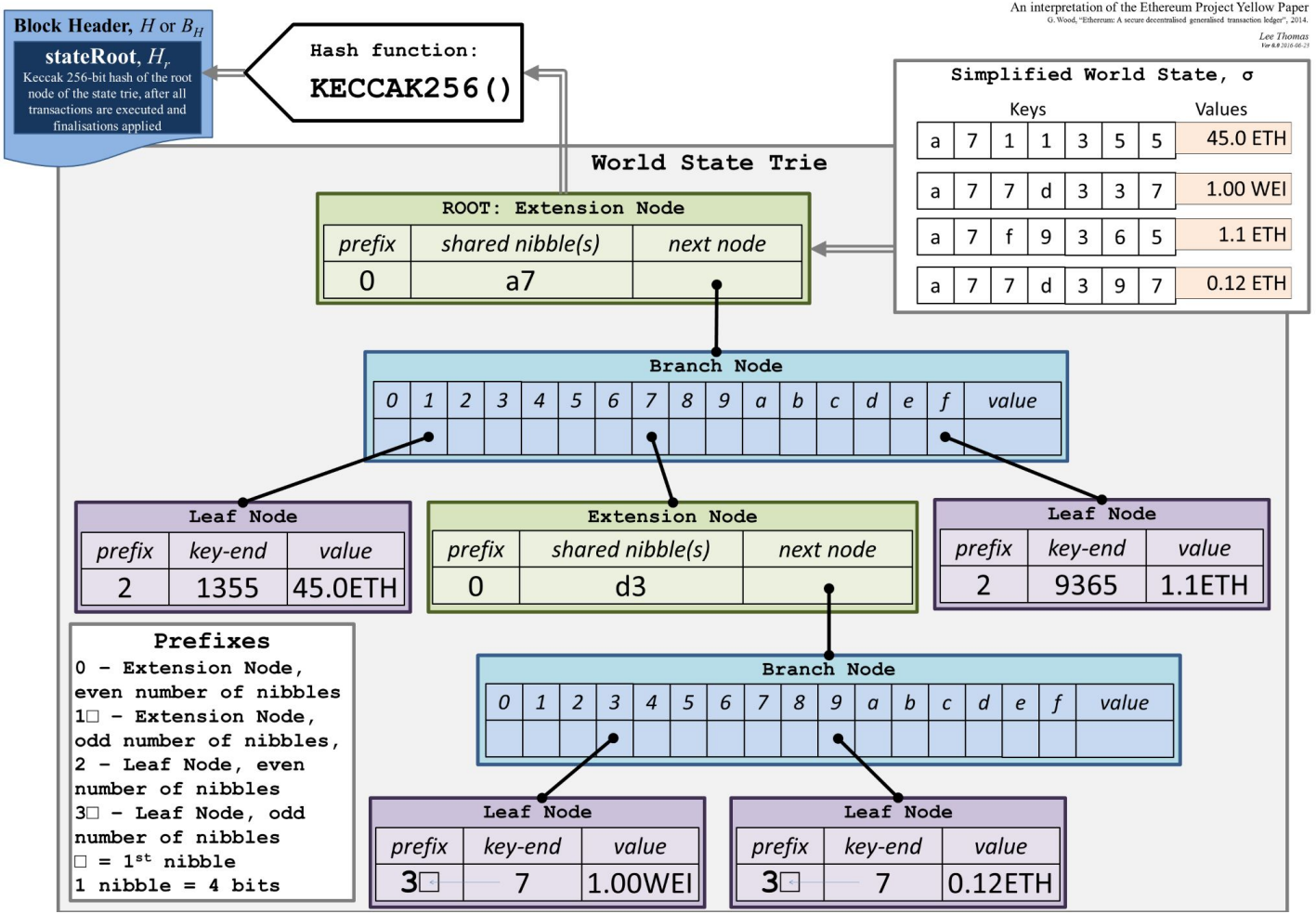
Courtesy : Takenobu T

Components of Ethereum Network - Transactions

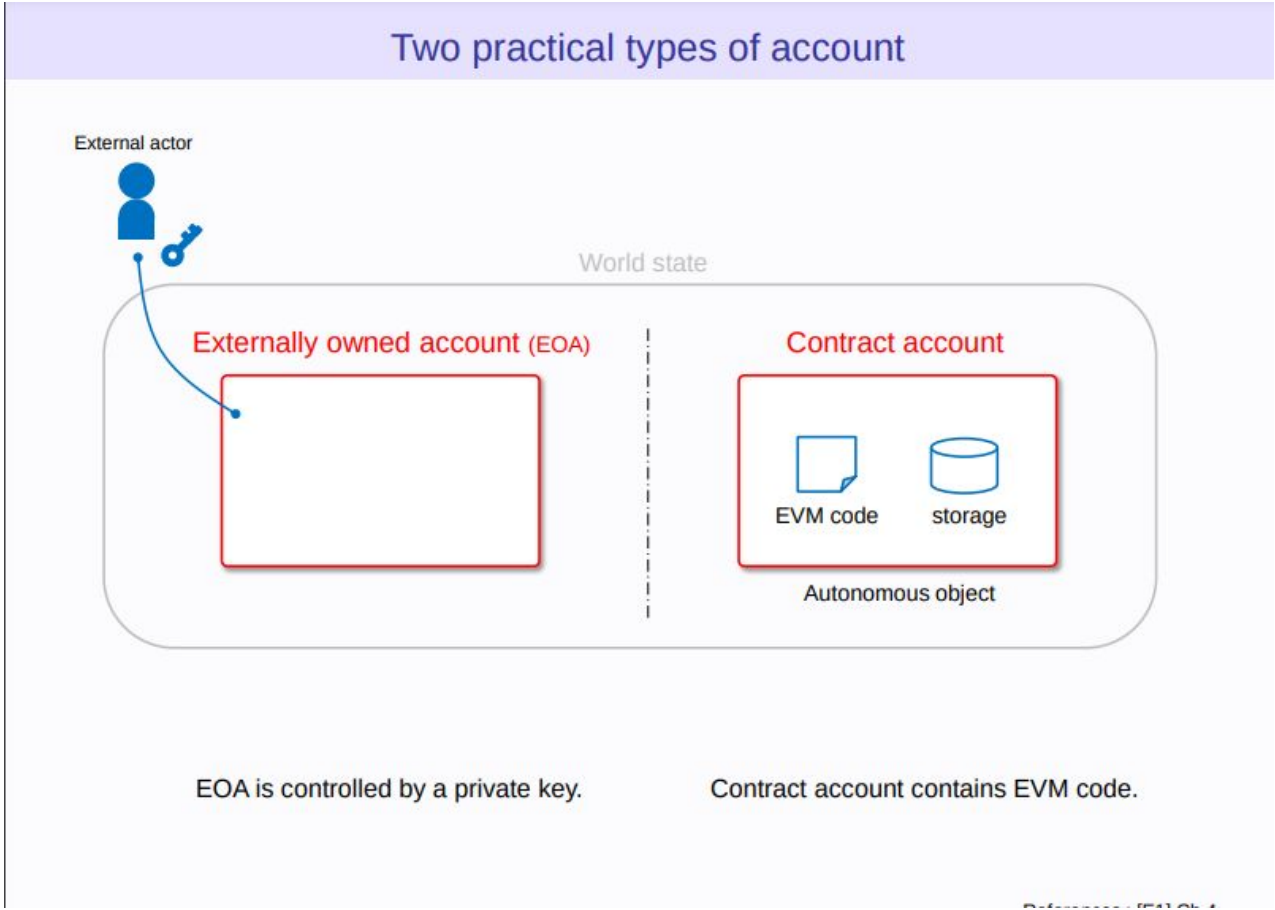


- `codeHash` – This hash refers to the *code* of an account on the Ethereum virtual machine (EVM). Contract accounts have code fragments programmed in that can perform different operations. This EVM code gets executed if the account gets a message call. It cannot be changed, unlike the other account fields. All such code fragments are contained in the state database under their corresponding hashes for later retrieval. This hash value is known as a codeHash. For externally owned accounts, the codeHash field is the hash of an empty string.

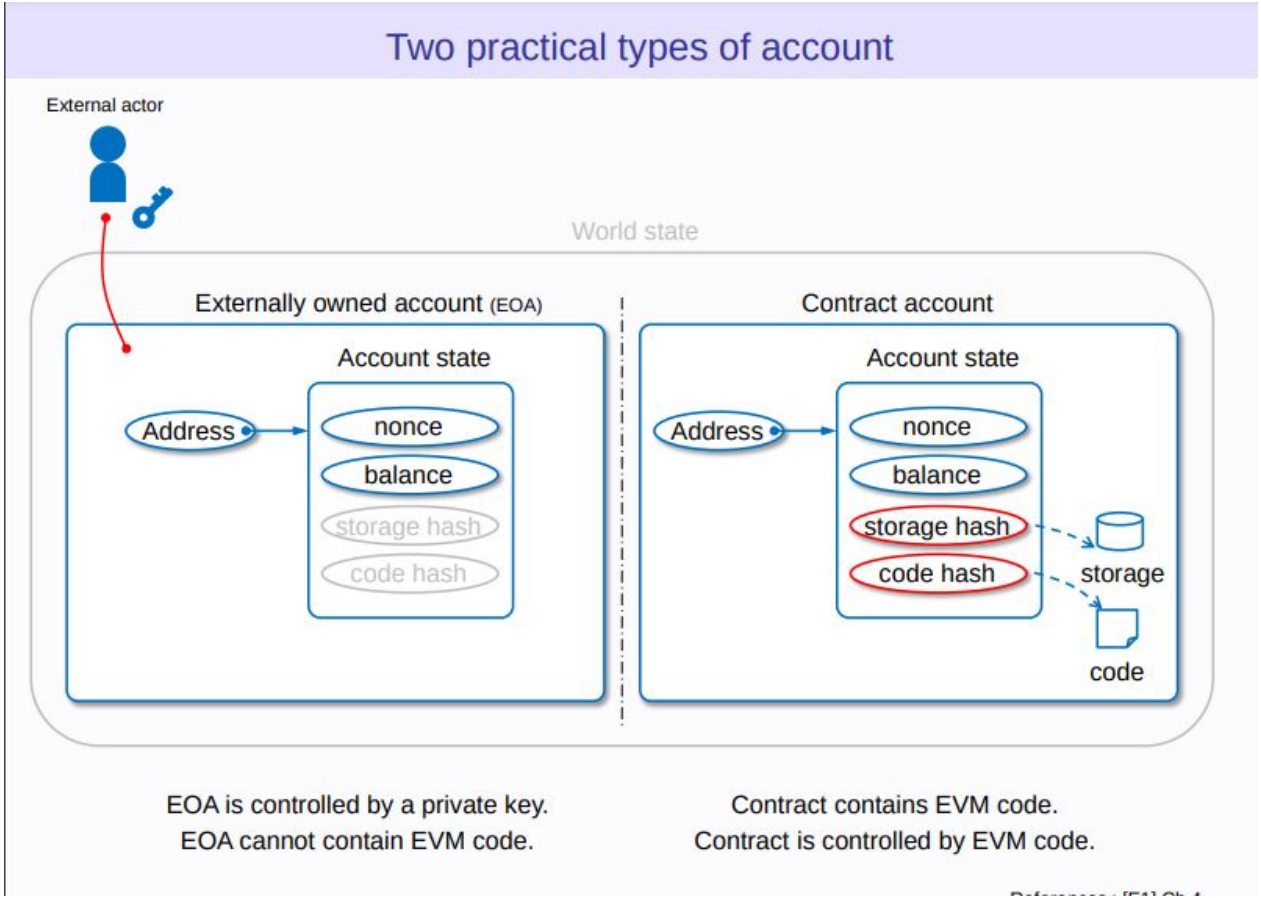
- `storageRoot` – Sometimes known as a storage hash. A 256-bit hash of the root node of a Merkle Patricia trie that encodes the storage contents of the account (a mapping between 256-bit integer values), encoded into the trie as a mapping from the Keccak 256-bit hash of the 256-bit integer keys to the RLP-encoded 256-bit integer values. This trie encodes the hash of the storage contents of this account, and is empty by default.


Ethereum Modified Merkle-Paricia-Trie System
An interpretation of the Ethereum Project Yellow Paper
G. Wood, "Ethereum: A secure decentralised generalised transaction ledger", 2014.
Lee Thomas
Ver 0.8 2016-06-25
Block Header, H or B_H
stateRoot, H_r
Keccak 256-bit hash of the root node of the state trie, after all transactions are executed and finalisations applied
Hash function:
KECCAK256()
Simplified World State, σ
Keys
Values
a 7 1 1 3 5 5 45.0 ETH
a 7 7 d 3 3 7 1.00 WEI
a 7 f 9 3 6 5 1.1 ETH
a 7 7 d 3 9 7 0.12 ETH
World State Trie
ROOT: Extension Node
prefix
shared nibble(s)
next node
0
a7
Branch Node
0 1 2 3 4 5 6 7 8 9 a b c d e f value
Leaf Node
prefix
key-end
value
2
1355
45.0ETH
Extension Node
prefix
shared nibble(s)
next node
0
d3
Leaf Node
prefix
key-end
value
2
9365
1.1ETH
Prefixes
0 - Extension Node, even number of nibbles
1□ - Extension Node, odd number of nibbles,
2 - Leaf Node, even number of nibbles
3□ - Leaf Node, odd number of nibbles
□ = 1st nibble
1 nibble = 4 bits
Branch Node
0 1 2 3 4 5 6 7 8 9 a b c d e f value
Leaf Node
prefix
key-end
value
3□
7
1.00WEI
Leaf Node
prefix
key-end
value
3□
7
0.12ETH

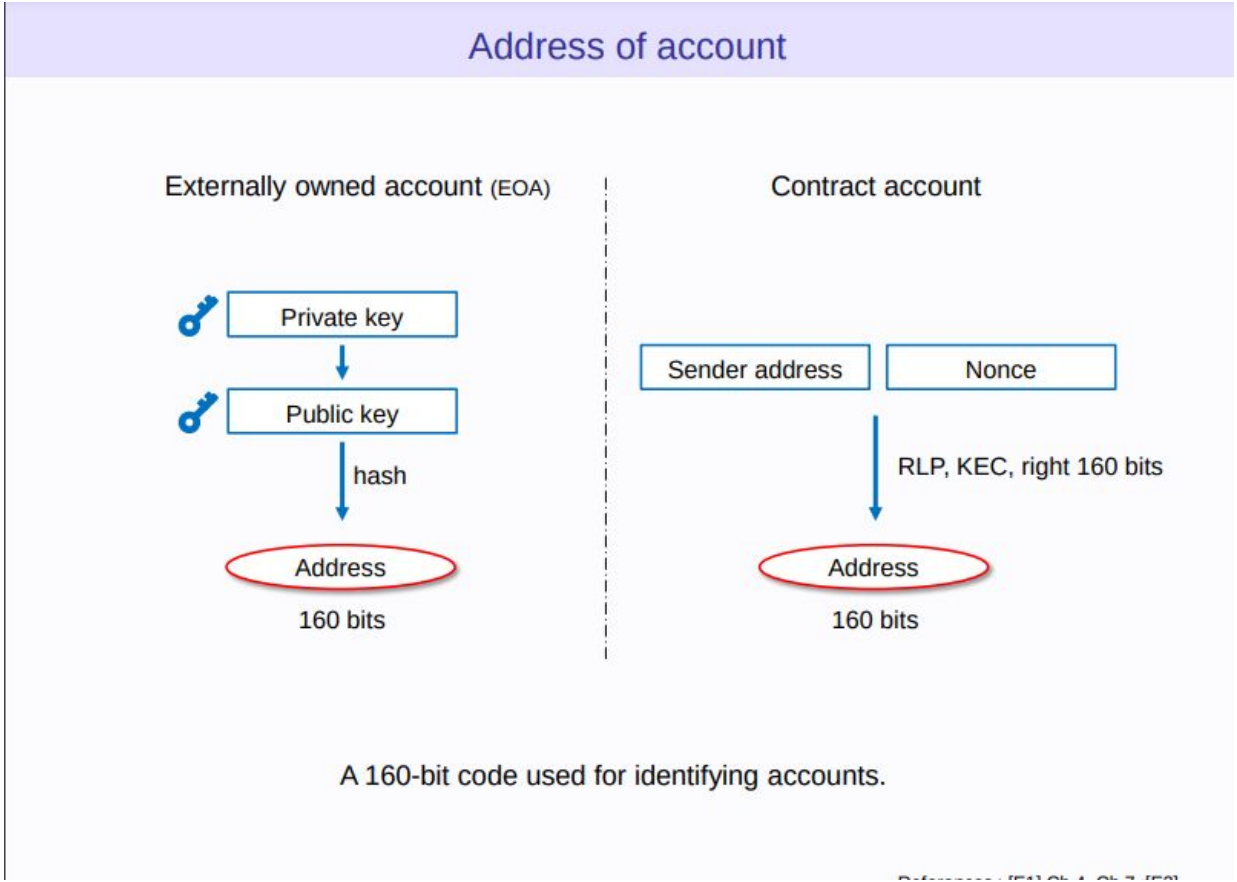
Components of Ethereum Network - Transactions



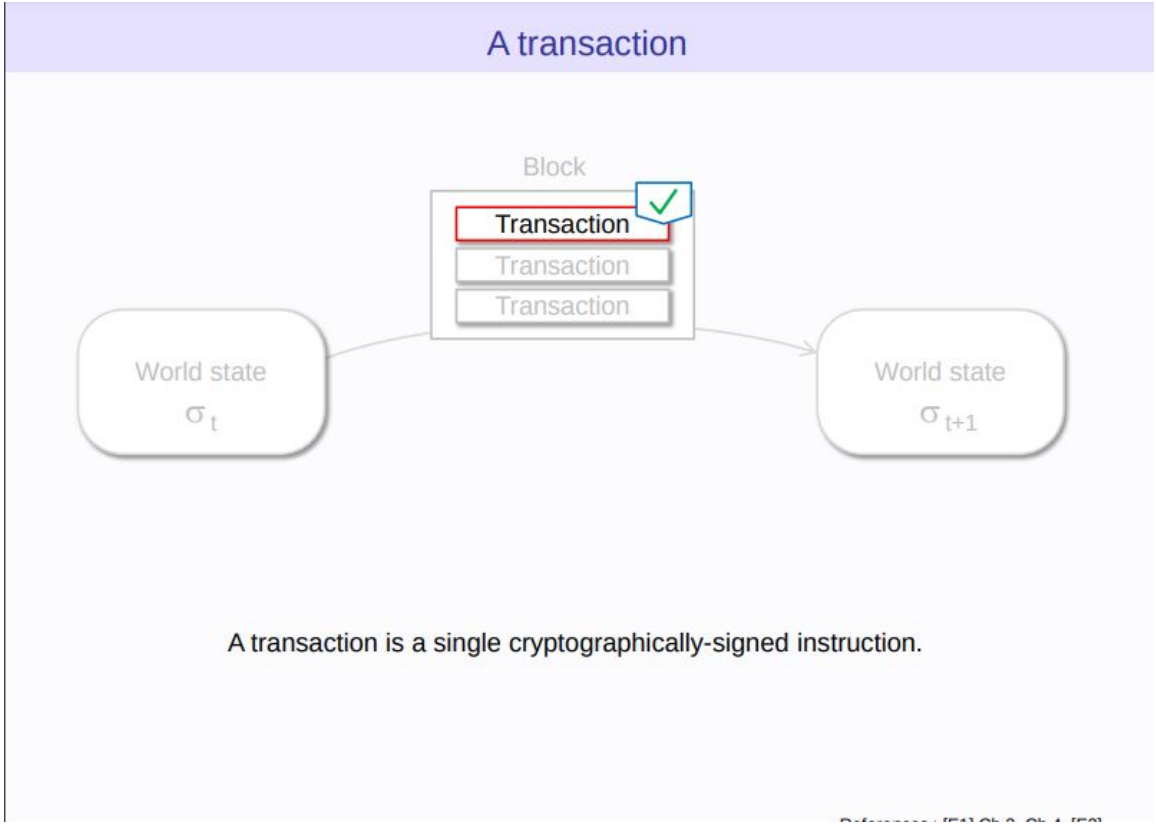
Components of Ethereum Network - Transactions



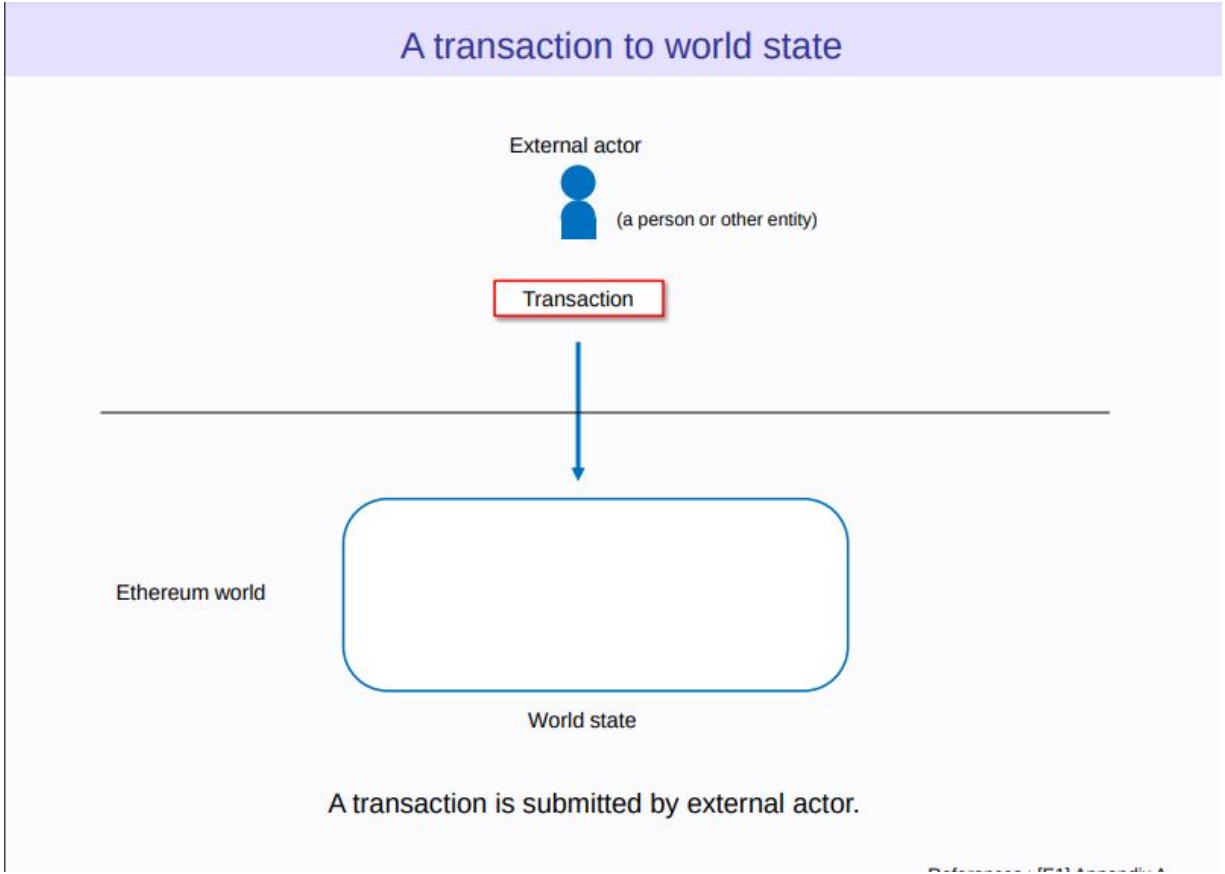
Components of Ethereum Network - Transactions



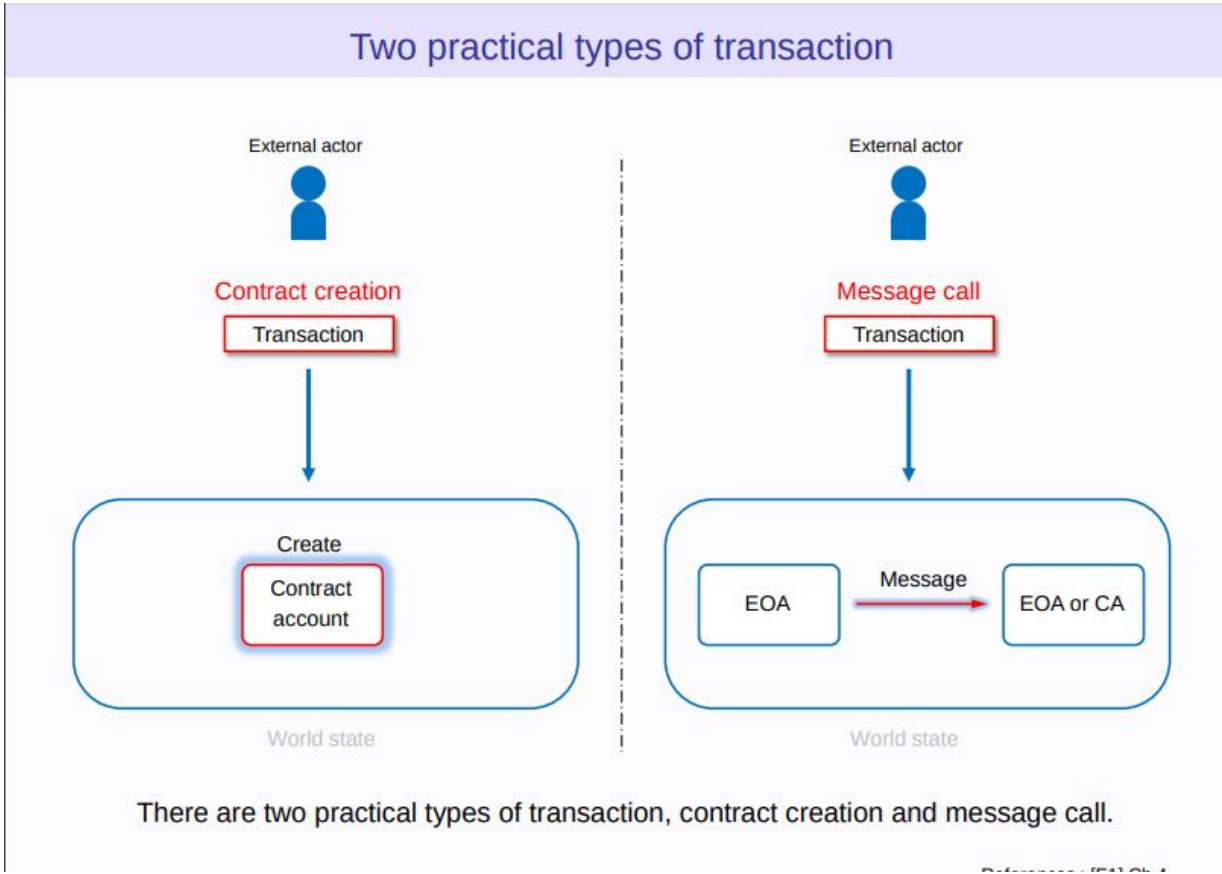
Components of Ethereum Network - Transactions



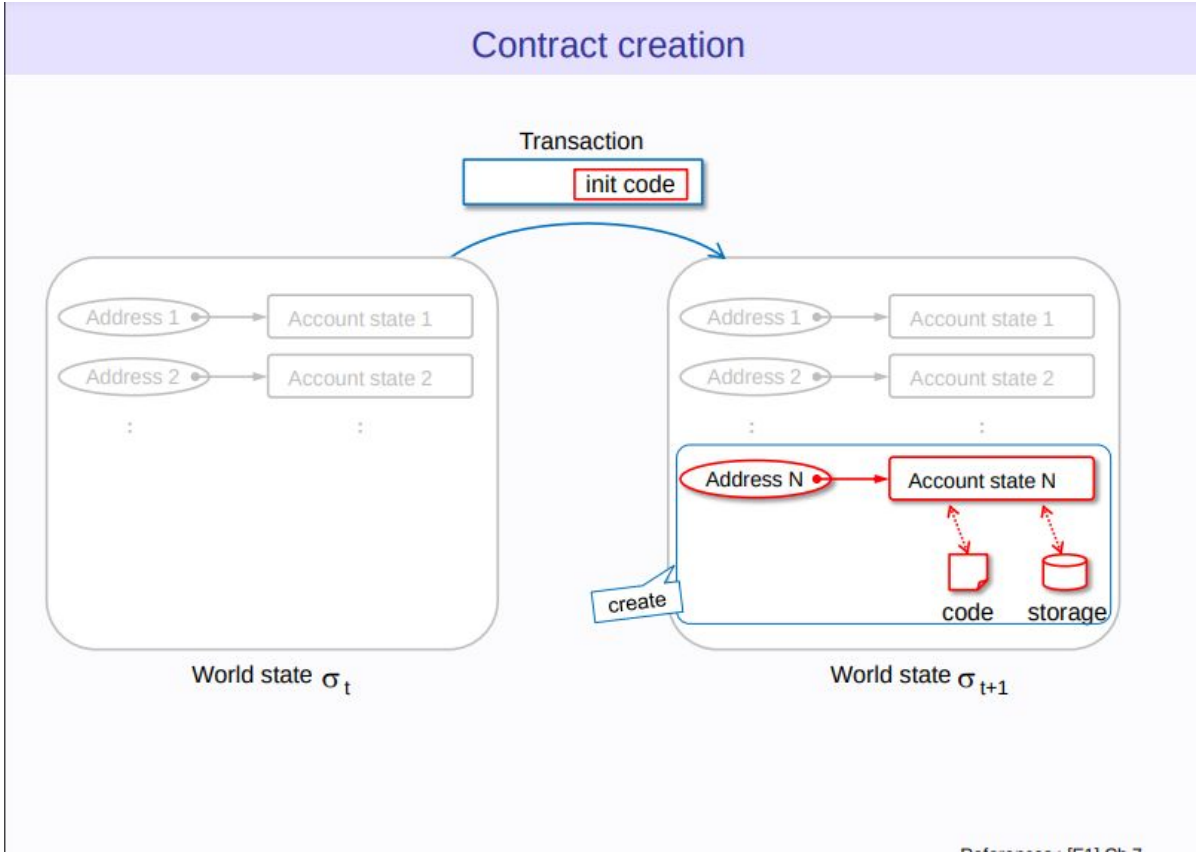
Components of Ethereum Network - Transactions



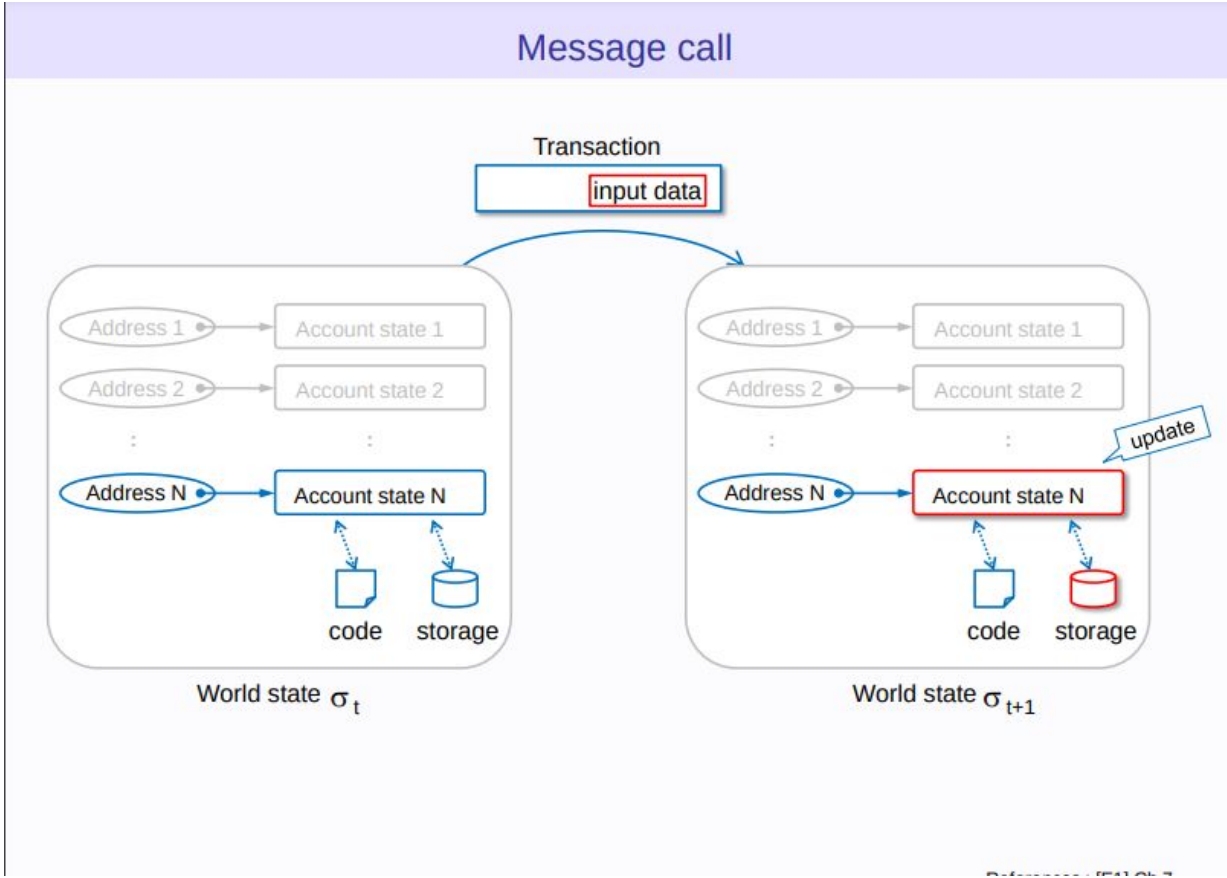
Components of Ethereum Network - Transactions



Components of Ethereum Network - Transactions

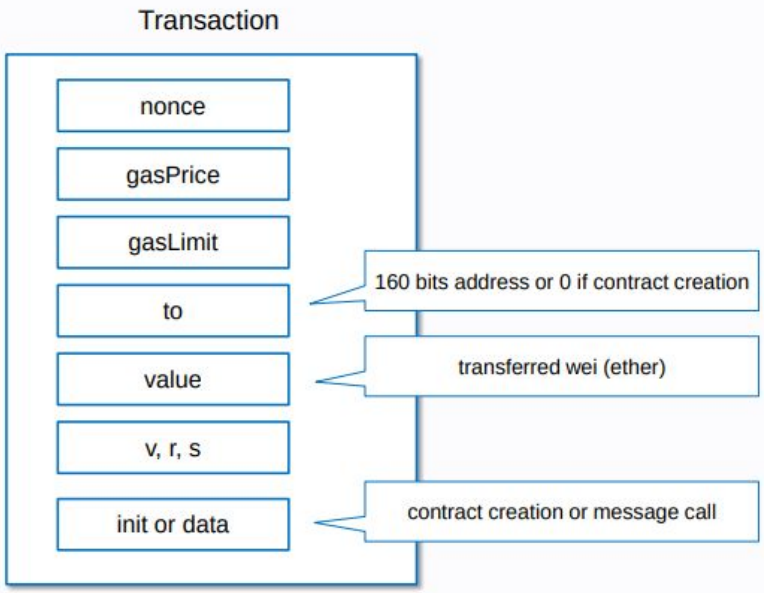


Components of Ethereum Network - Transactions



Components of Ethereum Network - Transactions

Field of a transaction



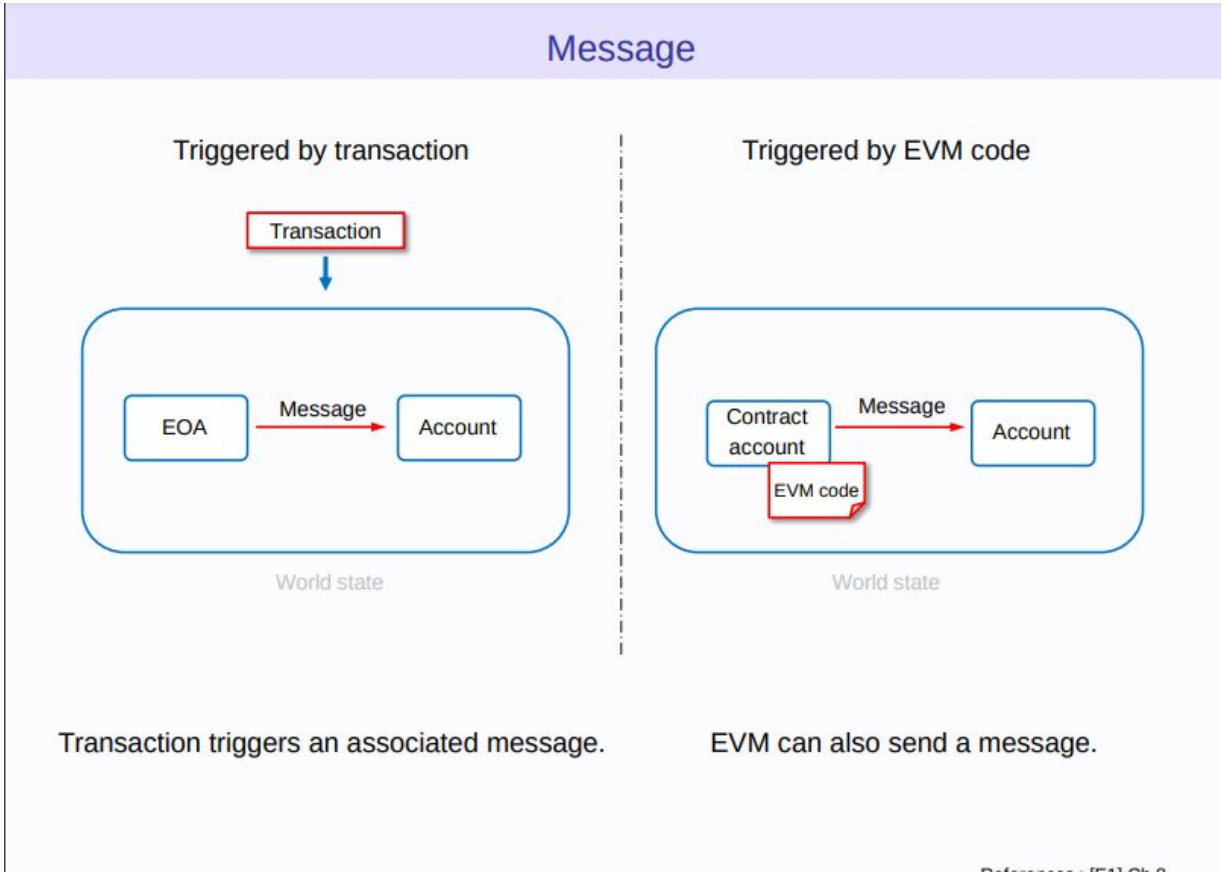
Components of Ethereum Network - Transactions

Message

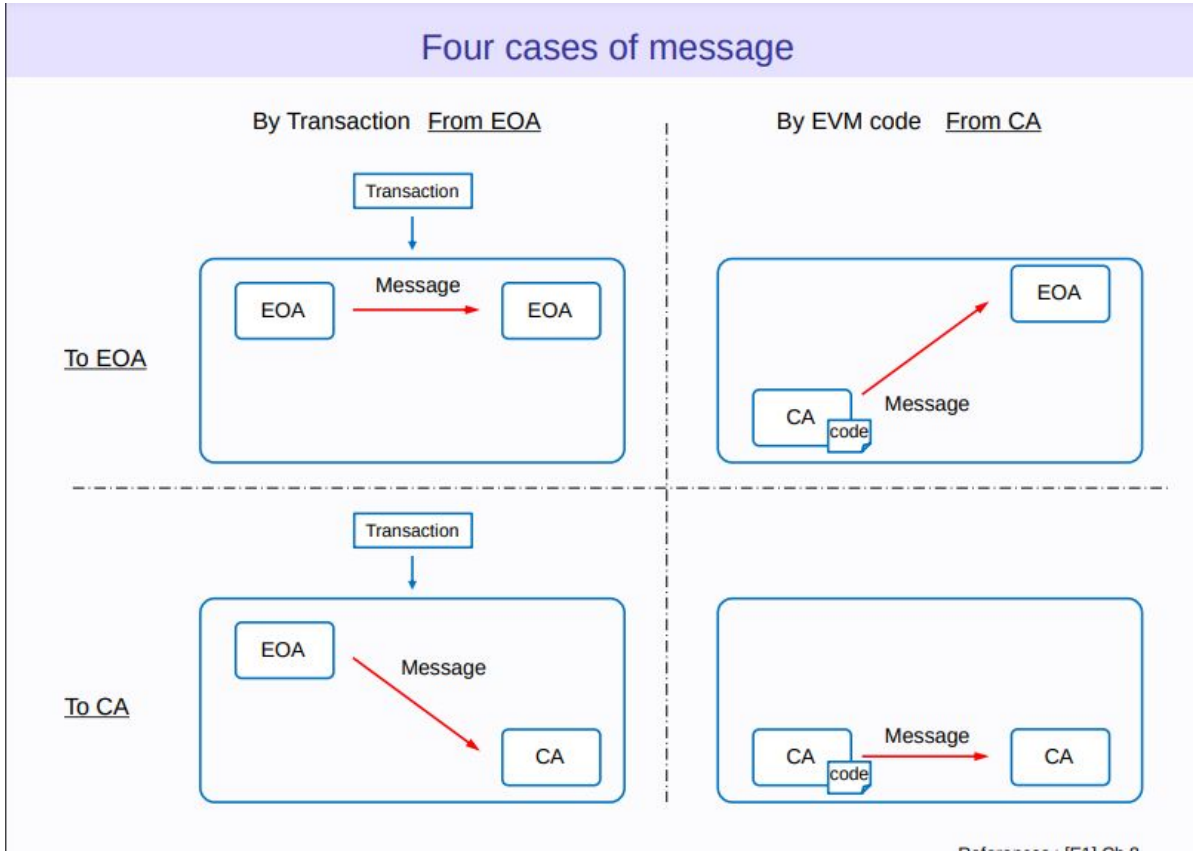
Message is passed between two Accounts.

Message is Data (as a set of bytes) and Value (specified as Ether) .

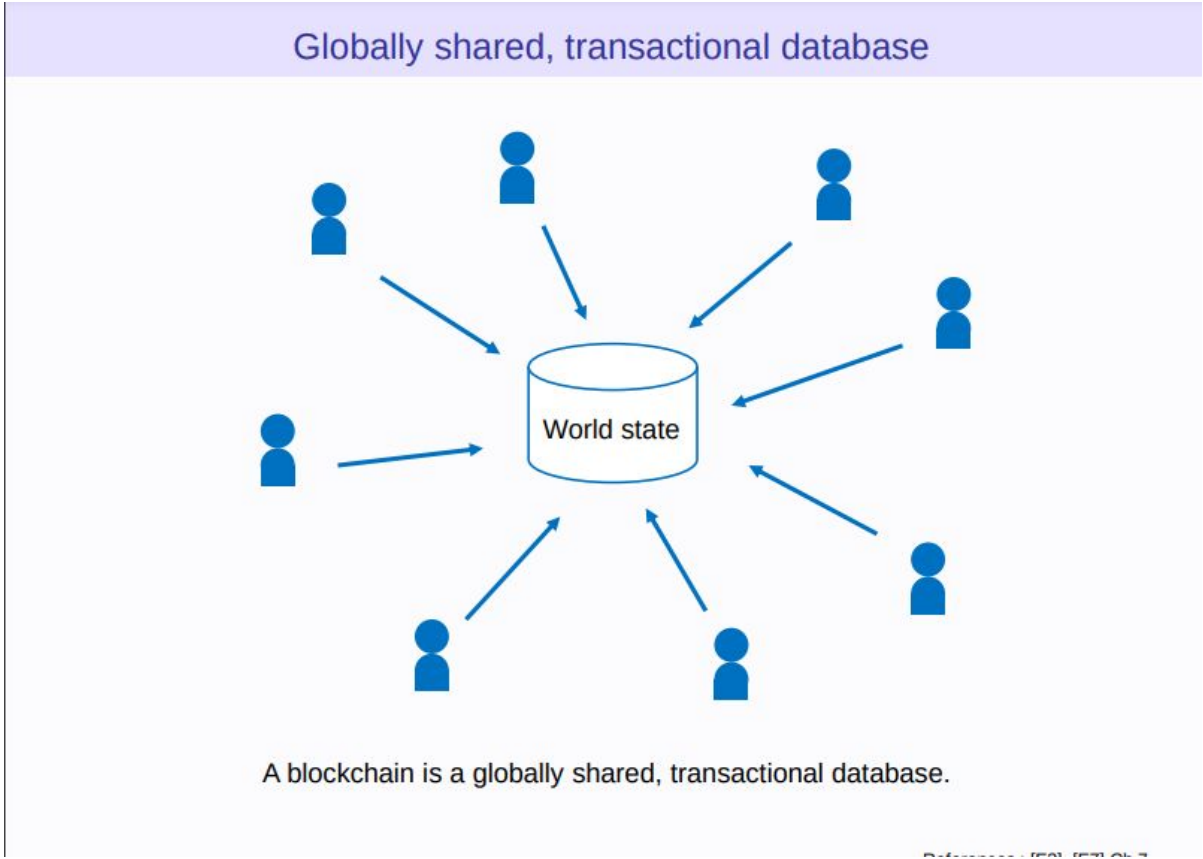
Components of Ethereum Network - Transactions



Components of Ethereum Network - Transactions

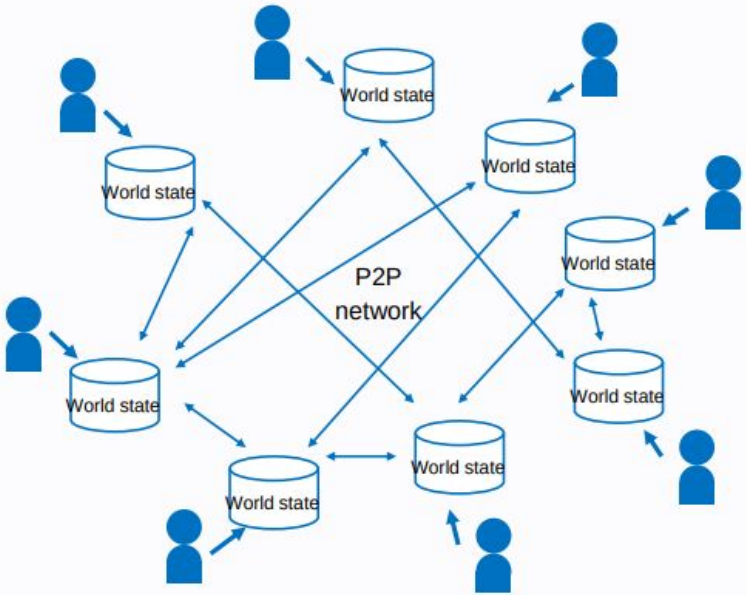


Components of Ethereum Network - Transactions



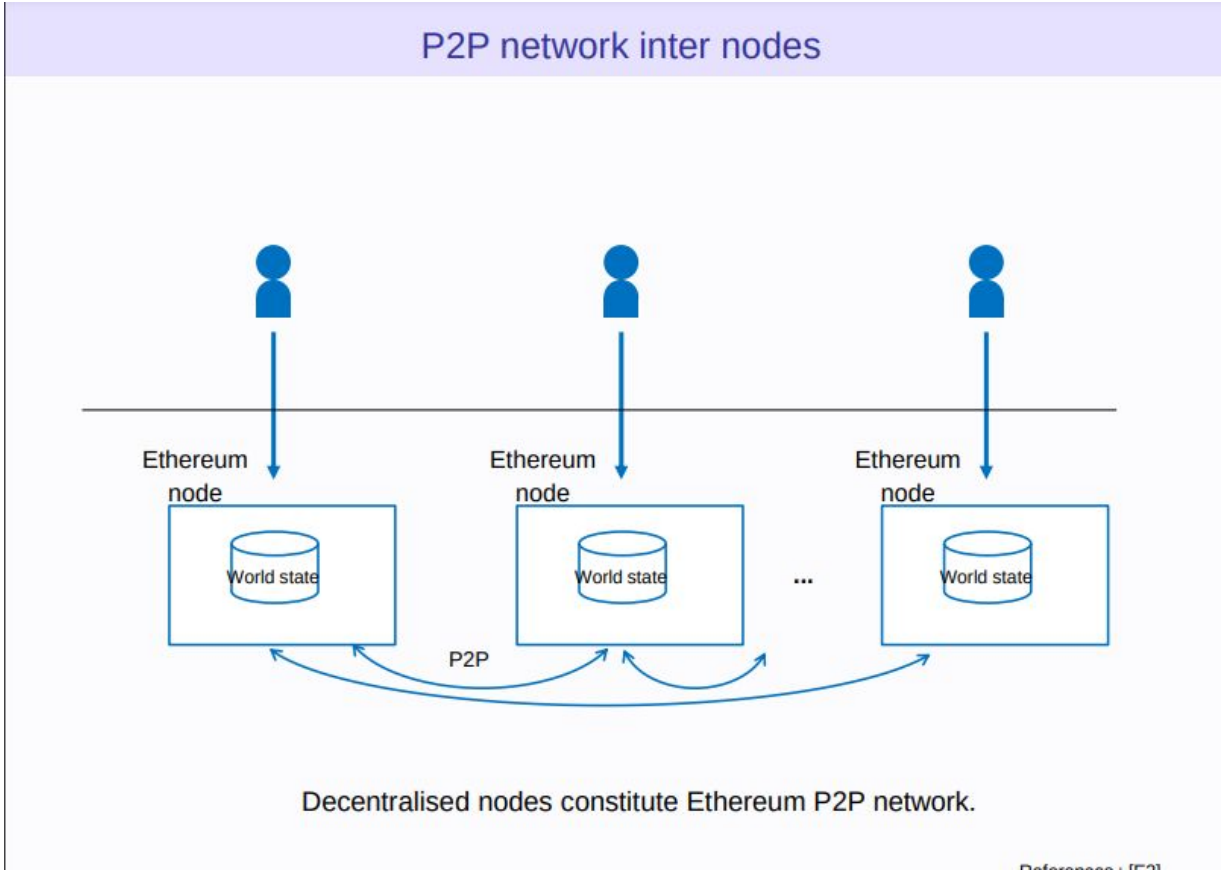
Components of Ethereum Network - Transactions

Decentralised database

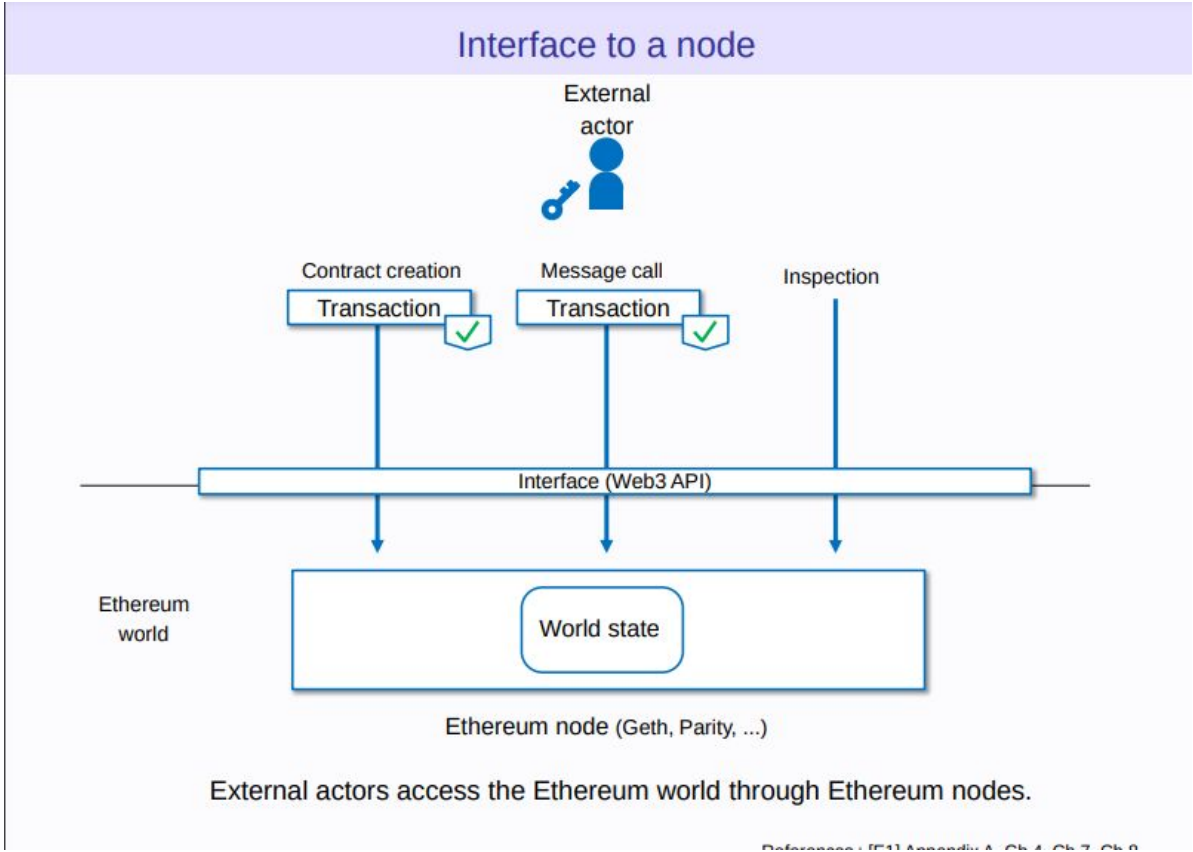


A blockchain is a globally shared, decentralised, transactional database.

Components of Ethereum Network - Transactions



Components of Ethereum Network - Transactions



Components of Ethereum Network - Transactions

Atomicity of transaction

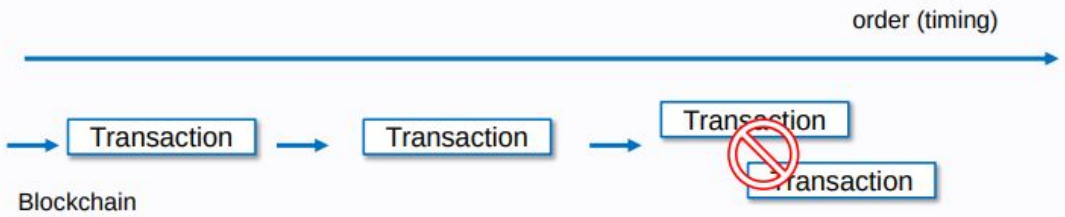
A transaction is an atomic operation. Can't divide or interrupt.

Transaction or Transaction

That is, All (complete done) or Nothing (zero effect).

Components of Ethereum Network - Transactions

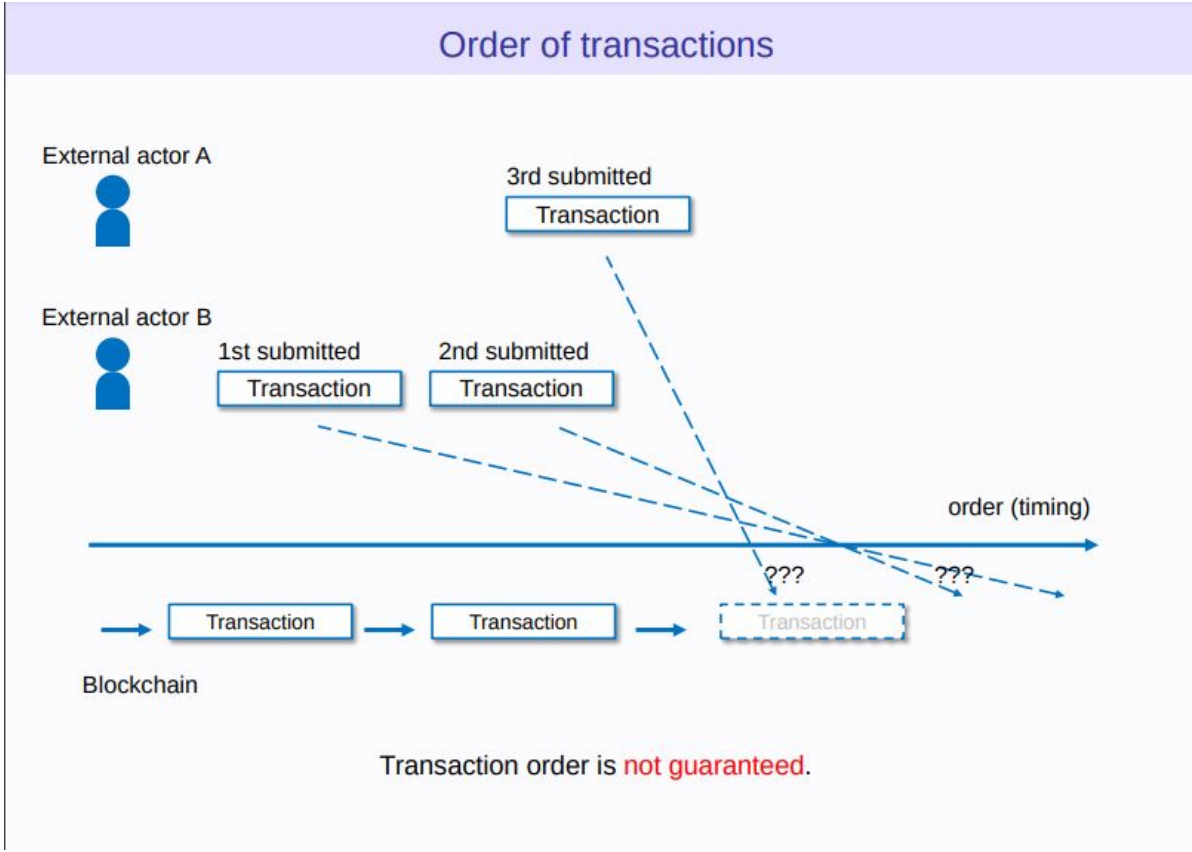
Order of transactions



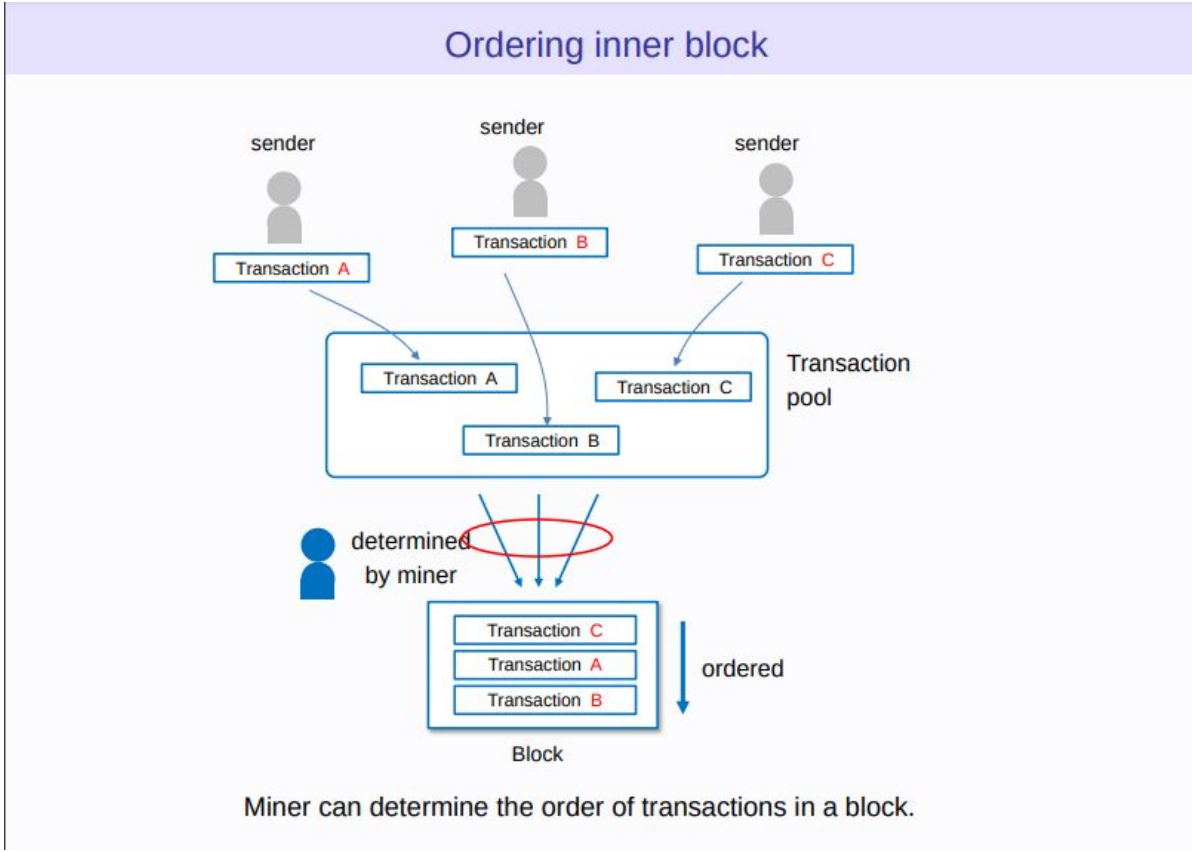
Transactions cannot be overlapped.

Transactions must be executed sequentially.

Components of Ethereum Network - Transactions

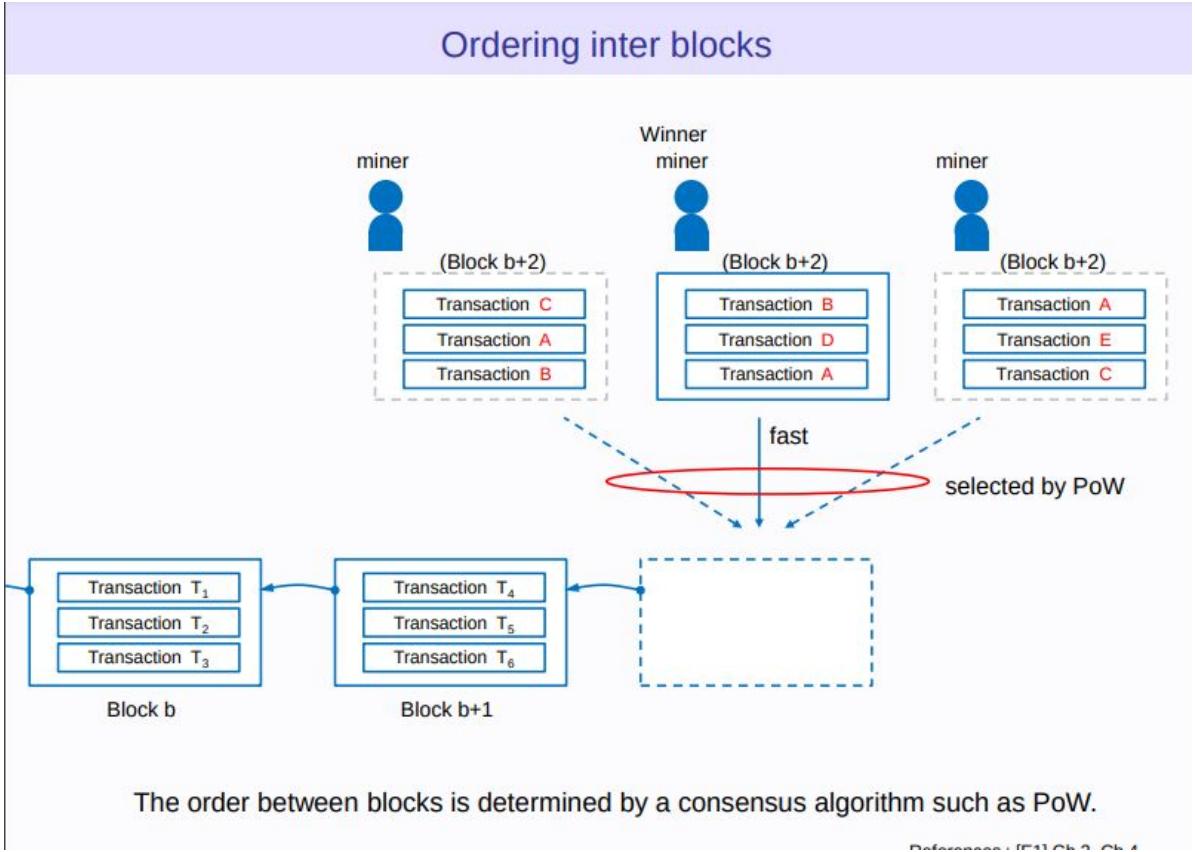


Components of Ethereum Network - Transactions

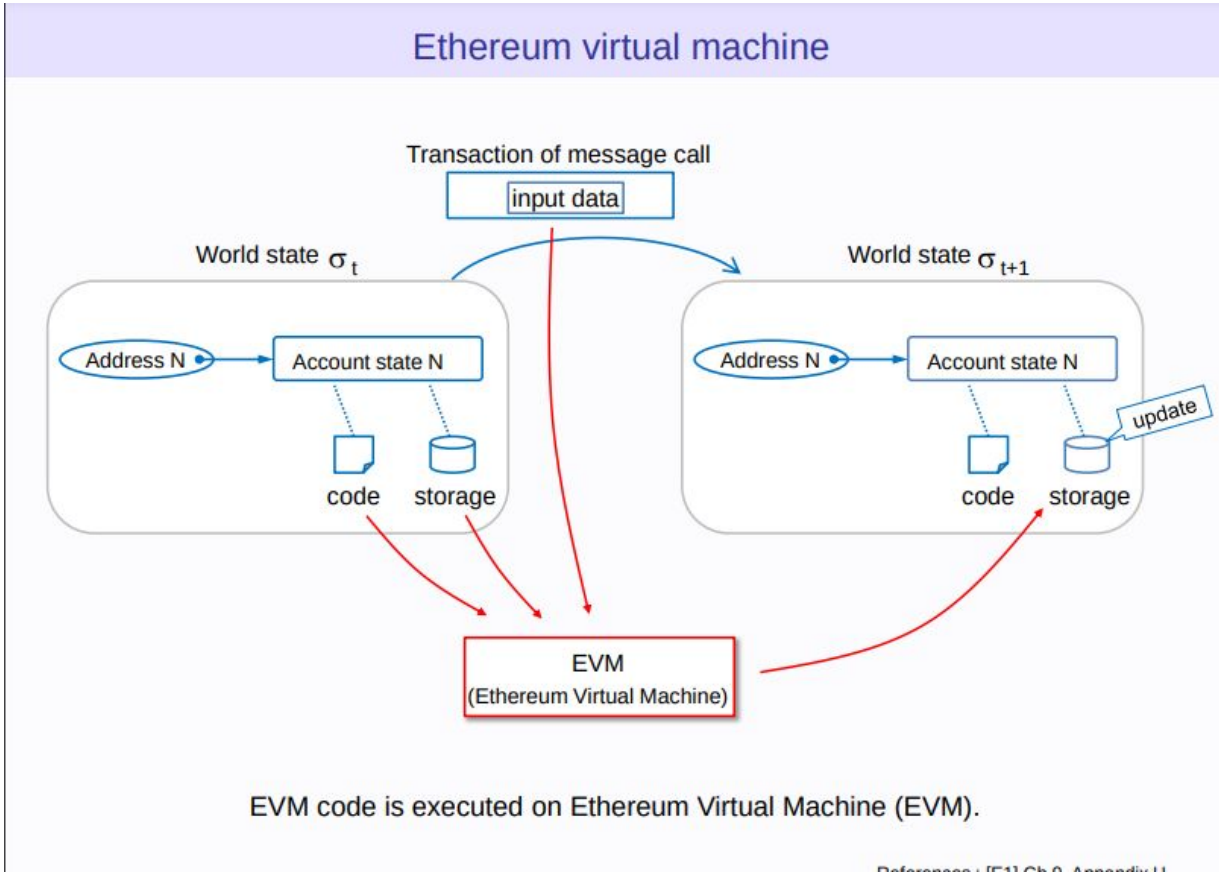


Courtesy : Takenobu T

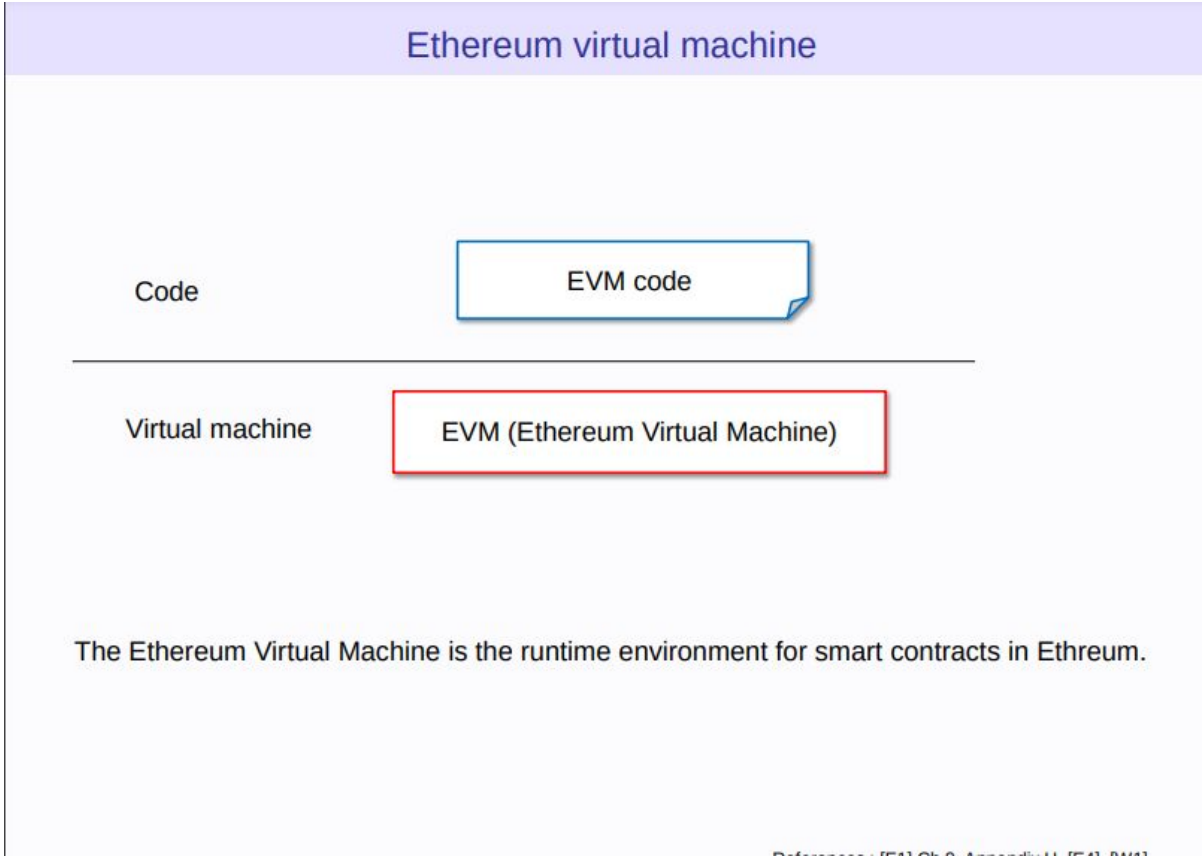
Components of Ethereum Network - Transactions



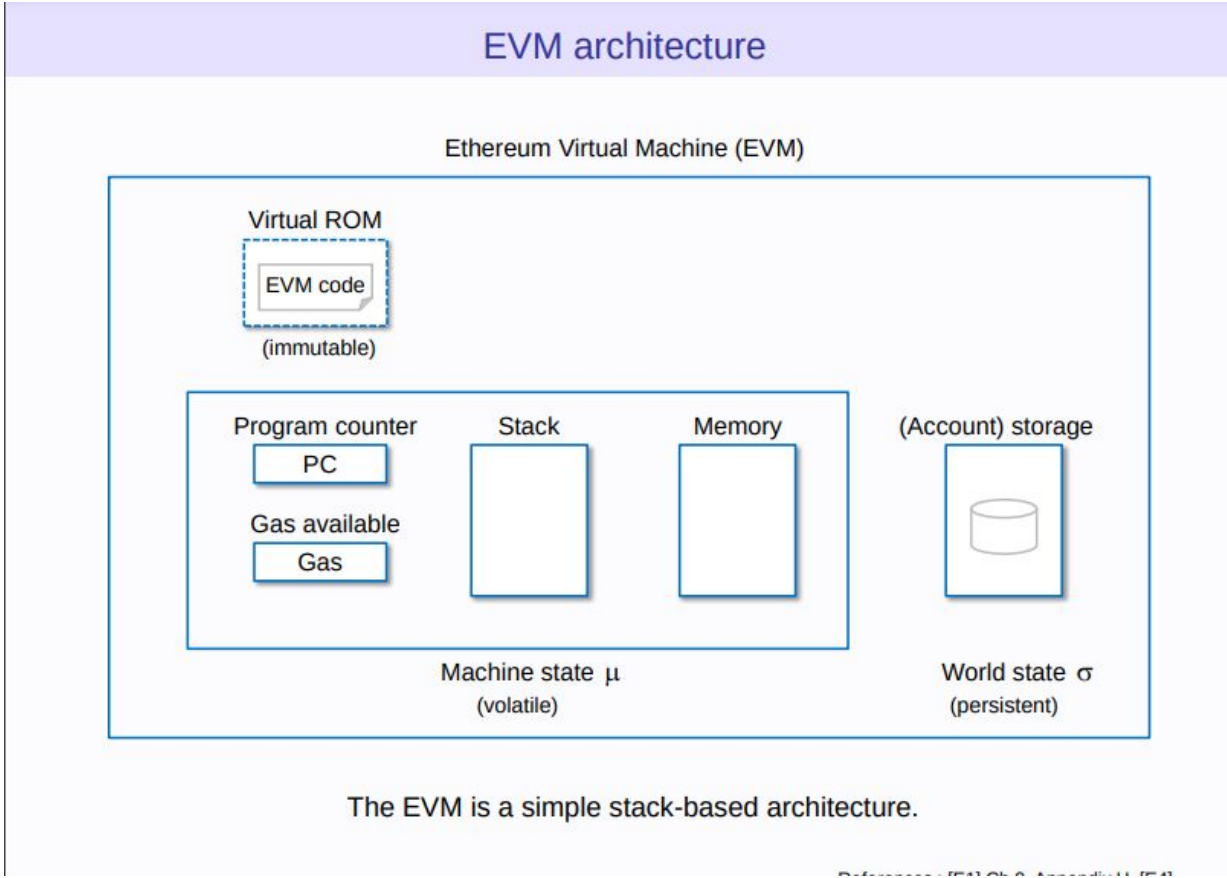
Components of Ethereum Network - Transactions & EVM



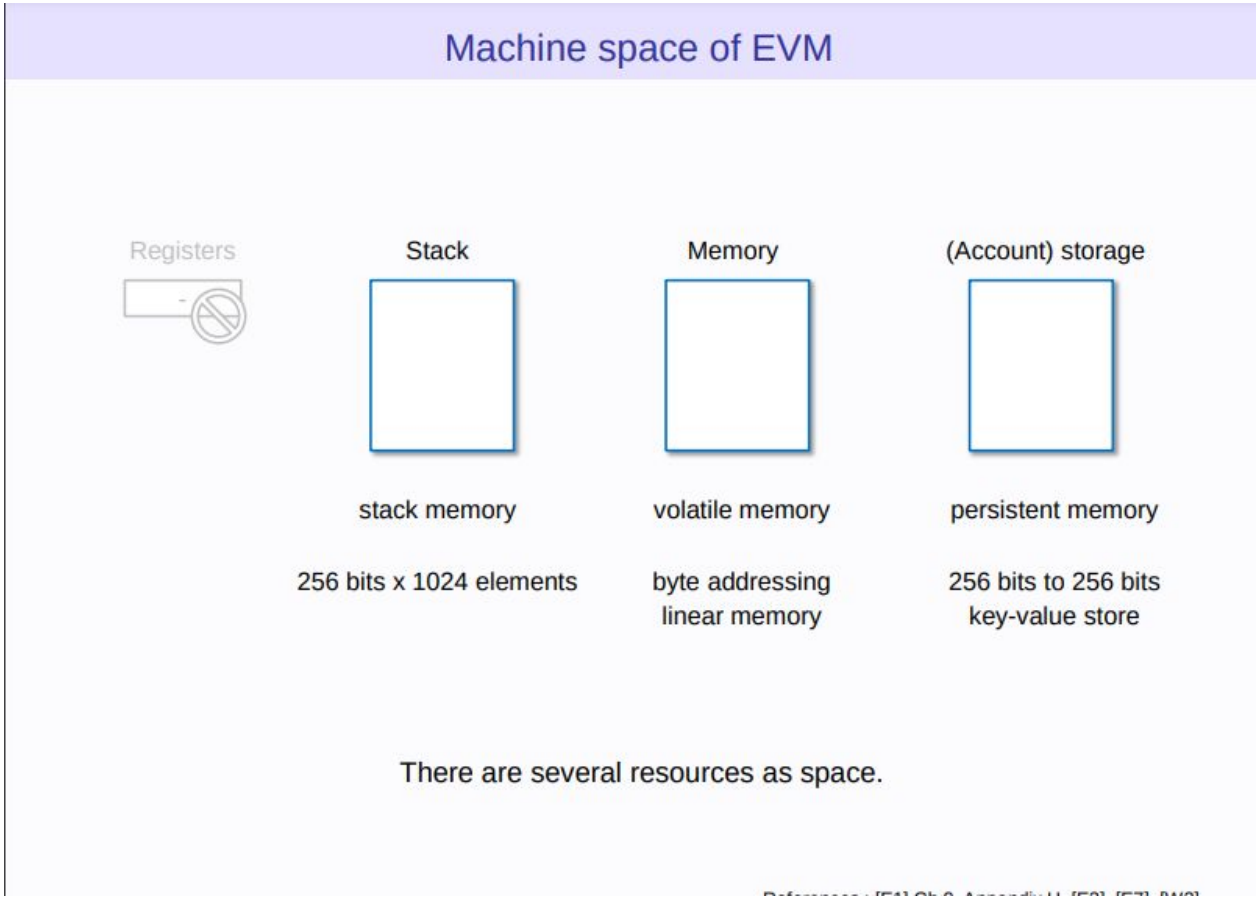
Components of Ethereum Network - EVM



Components of Ethereum Network - EVM Architecture

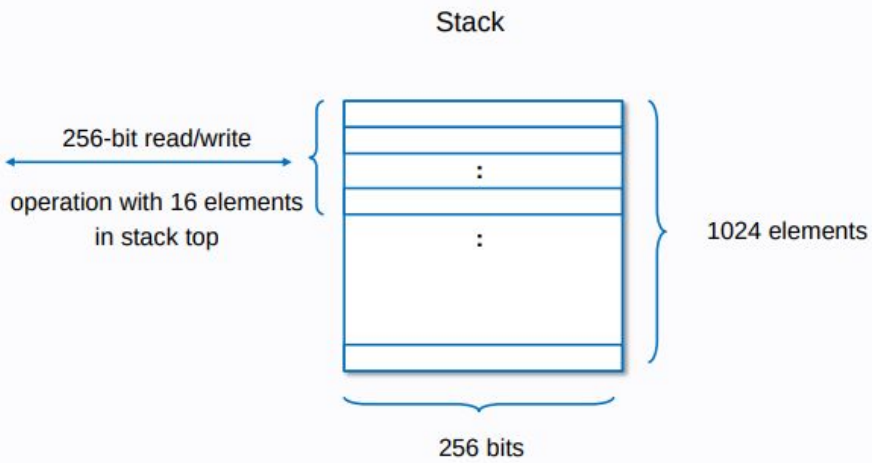


Components of Ethereum Network - EVM Architecture



Components of Ethereum Network - EVM Architecture

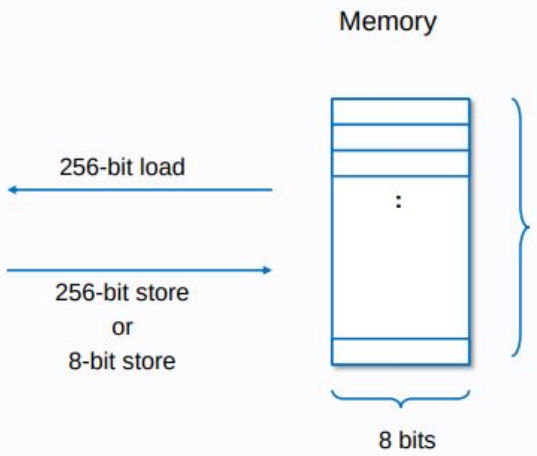
Stack



All operation are performed on the stack.
Access with many instructions such as PUSH/POP/COPY/SWAP, ...

Components of Ethereum Network - EVM Architecture

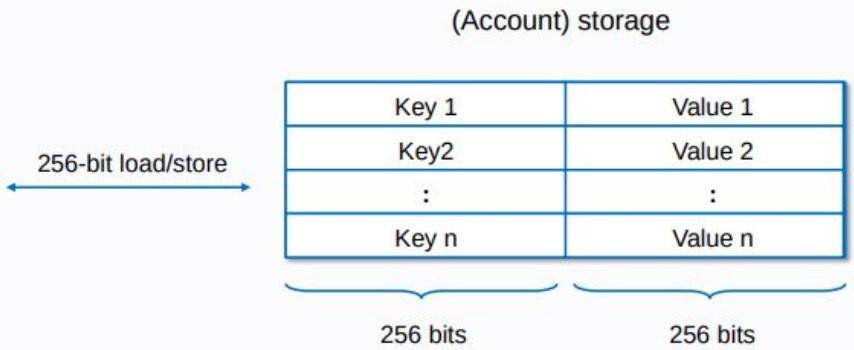
Memory



Memory is linear and can be addressed at byte level.
Access with MSTORE/MSTORE8/MLOAD instructions.
All locations in memory are well-defined initially as zero.

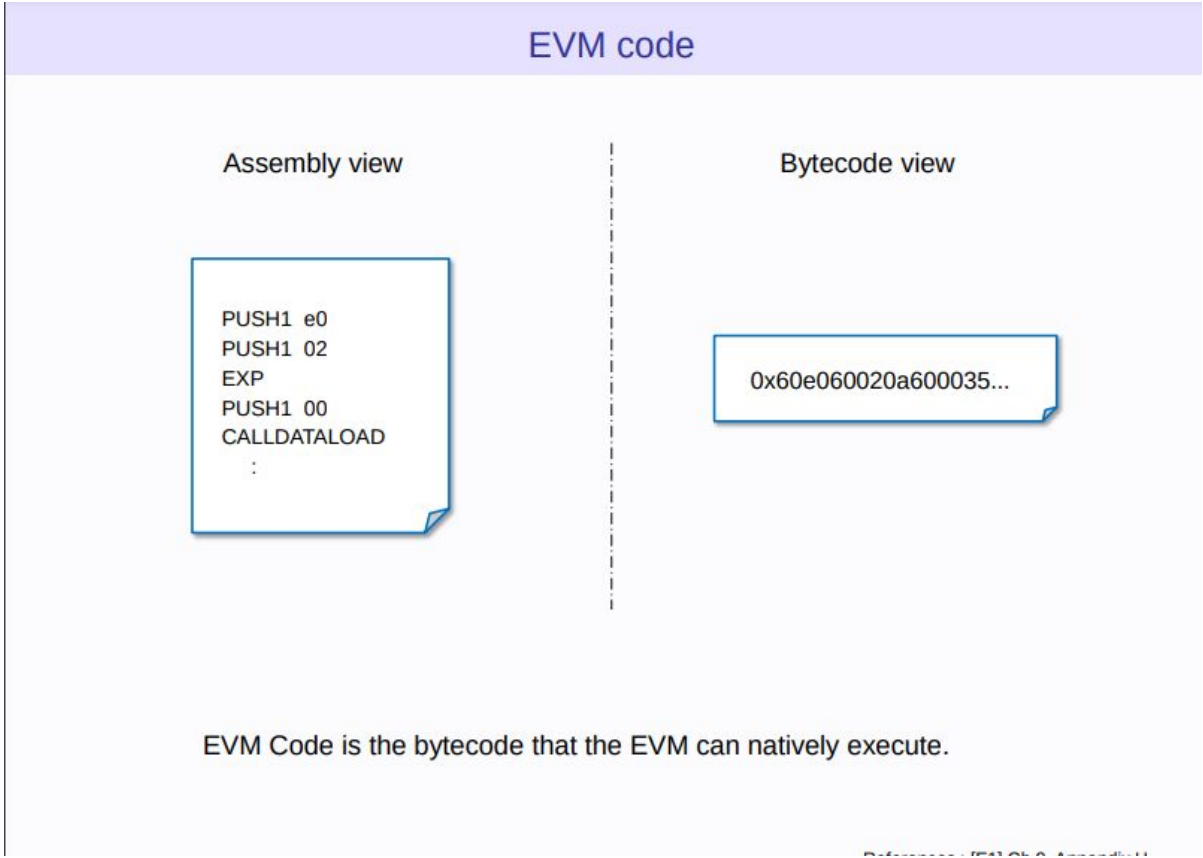
Components of Ethereum Network - EVM Architecture

Account storage

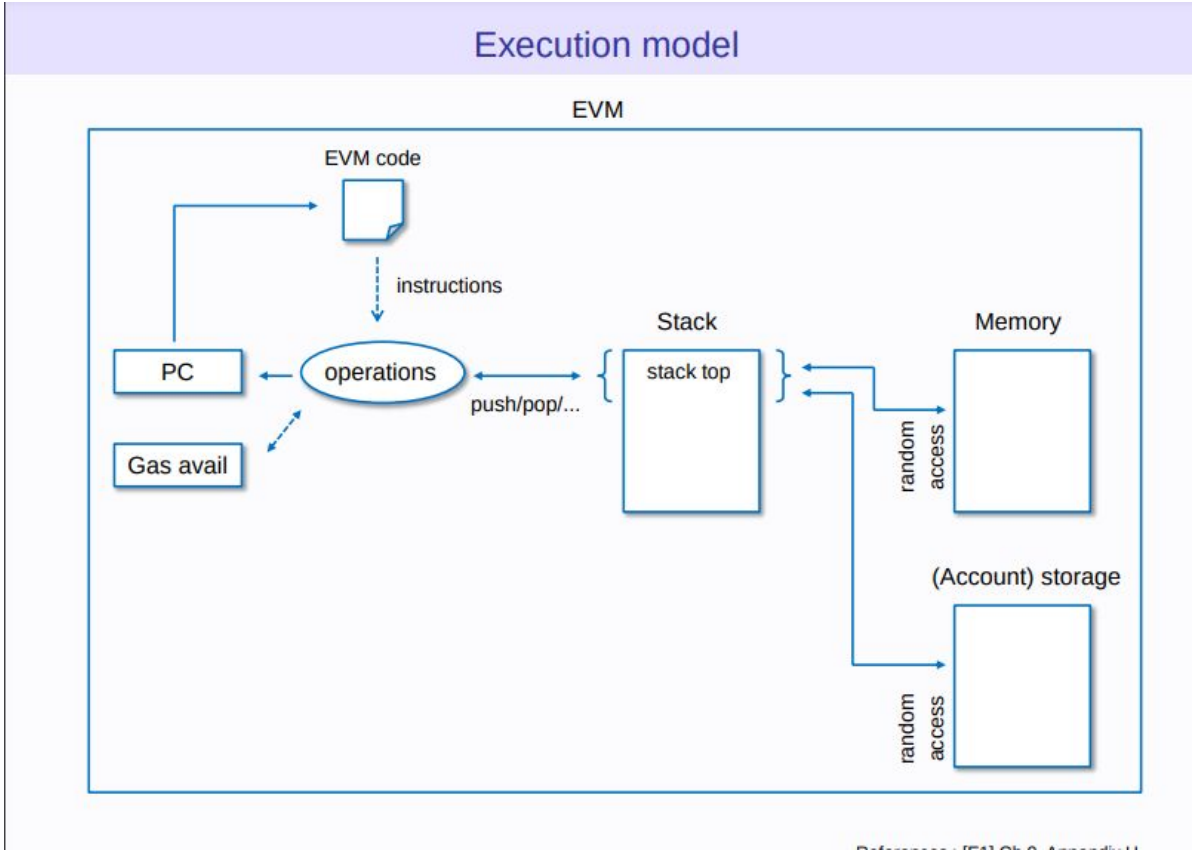


Storage is a key-value store that maps 256-bit words to 256-bit words.
Access with SSTORE/SLOAD instructions.
All locations in storage are well-defined initially as zero.

Components of Ethereum Network - EVM Architecture

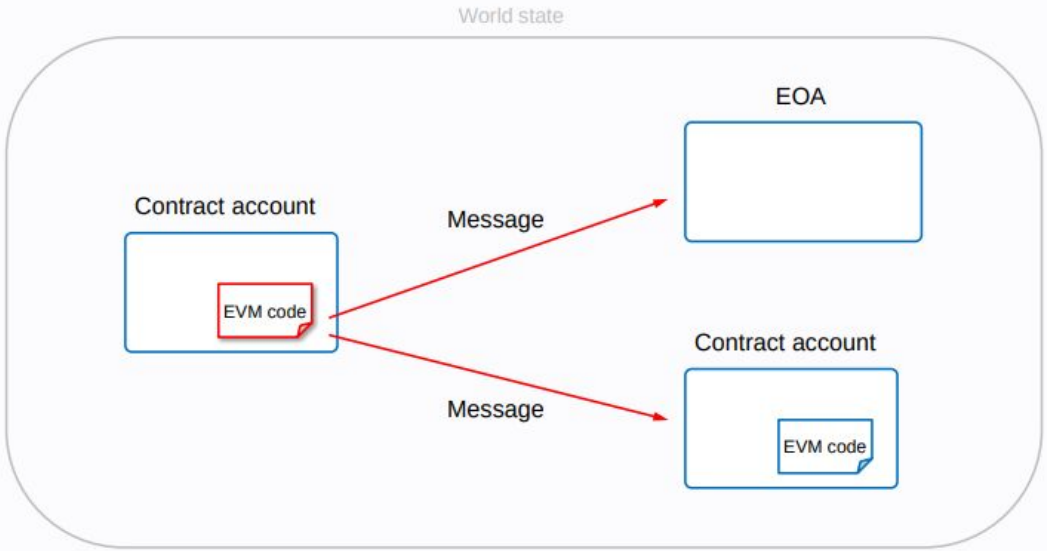


Components of Ethereum Network - EVM Architecture



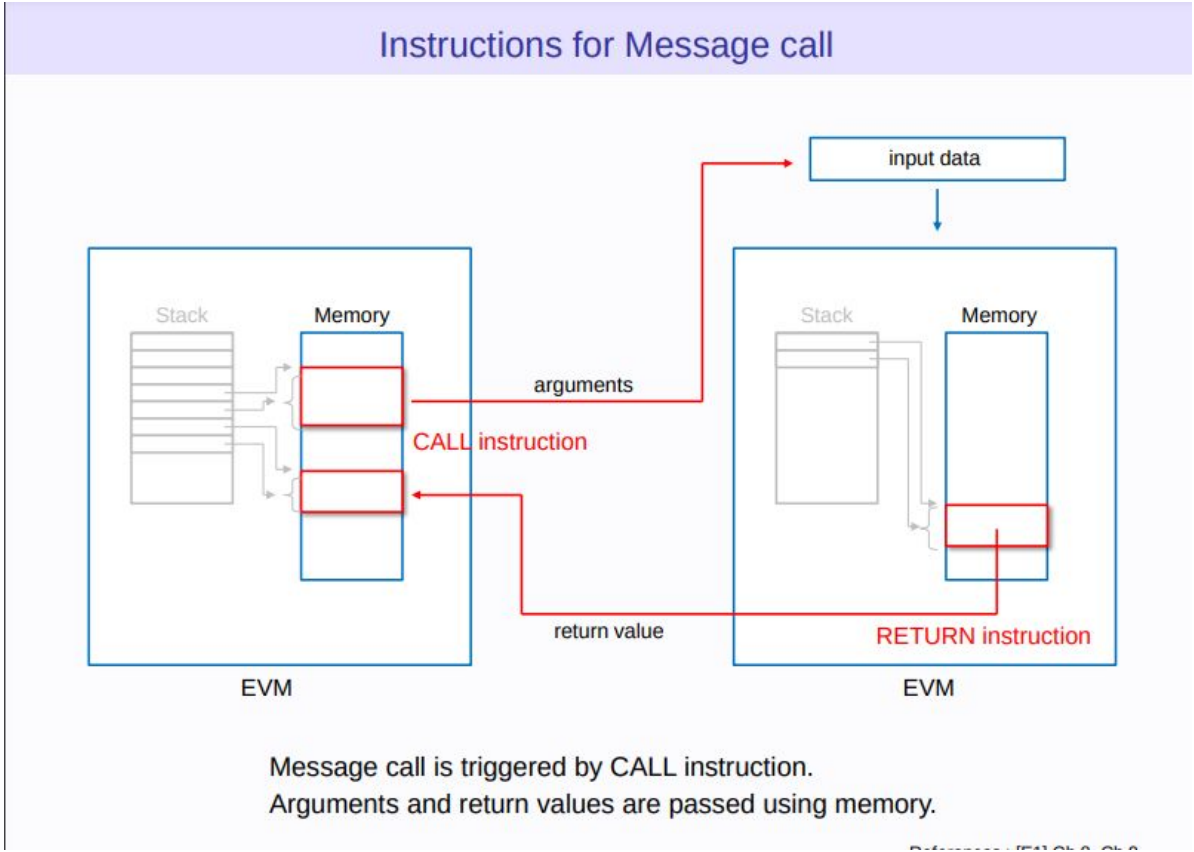
Components of Ethereum Network - Transactions

Message call

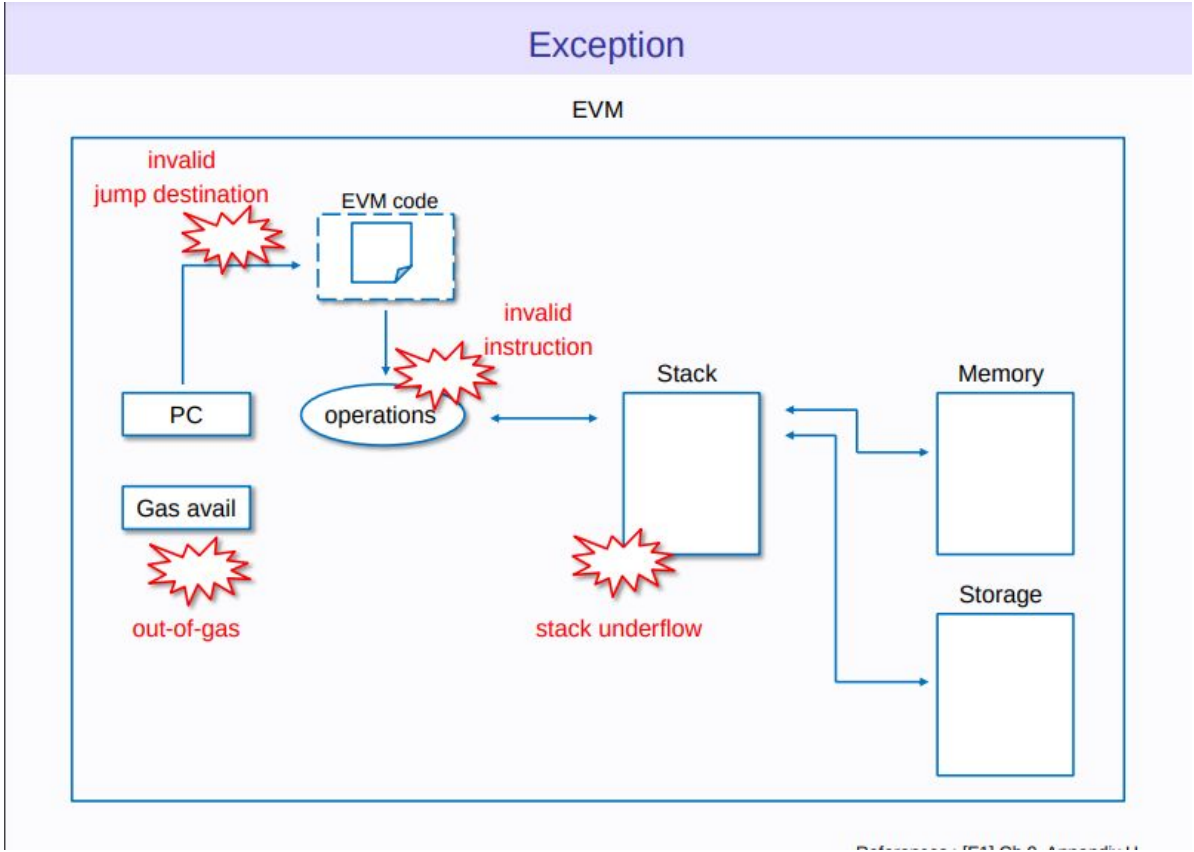


EVM can send a message to other account.
The depth of message call is limited to less than 1024 levels.

Components of Ethereum Network - Transactions

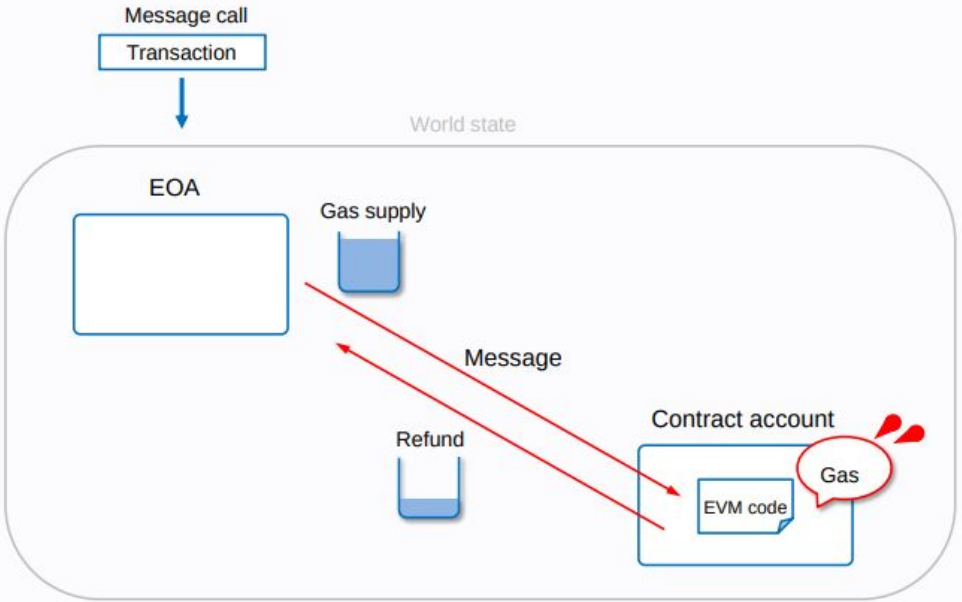


Components of Ethereum Network - Transactions



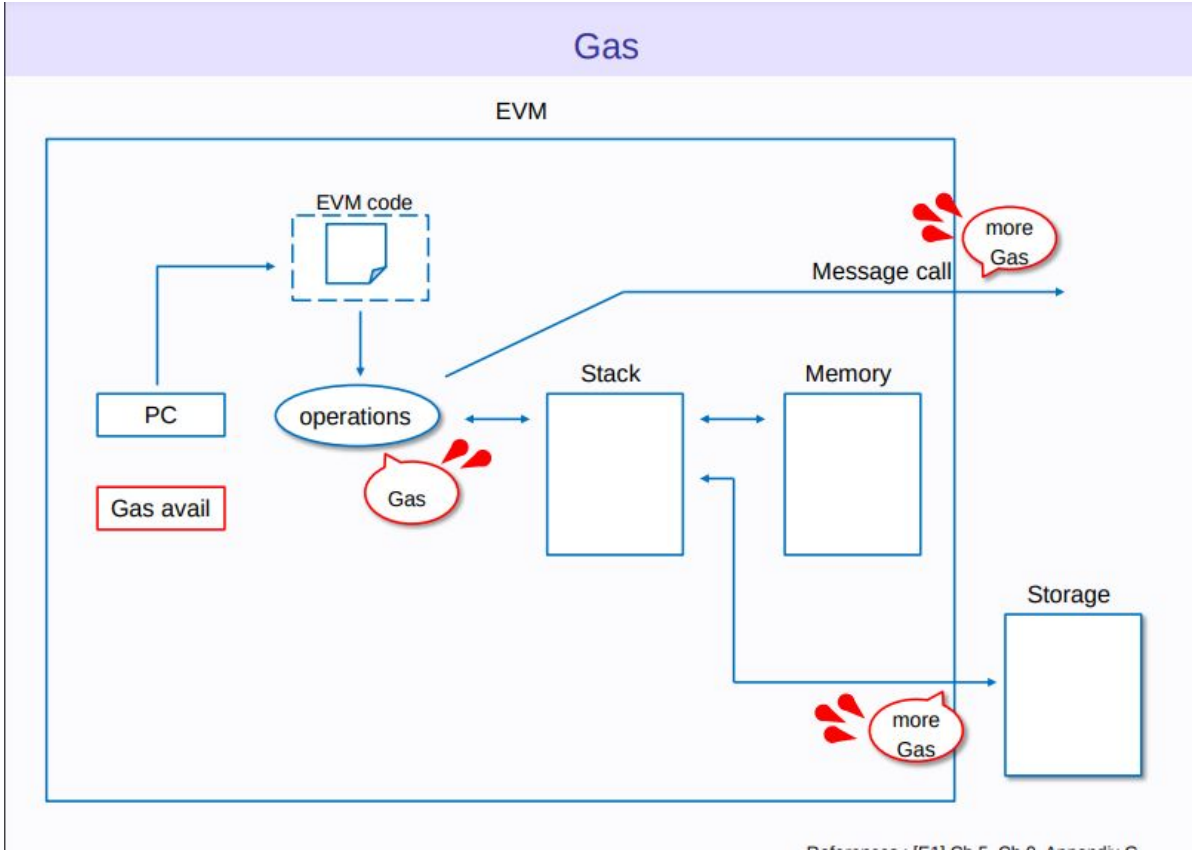
Components of Ethereum Network - Transactions

Gas and fee



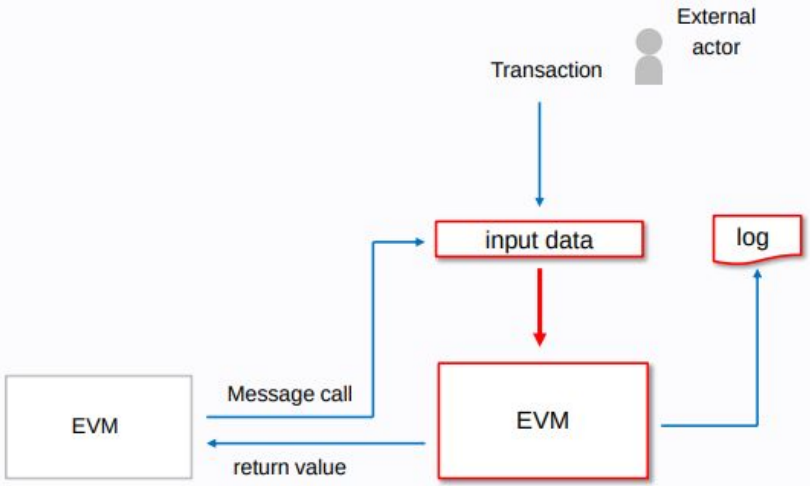
All programmable computation in Ethereum is subject to fees (denominated in gas).

Components of Ethereum Network - Transactions



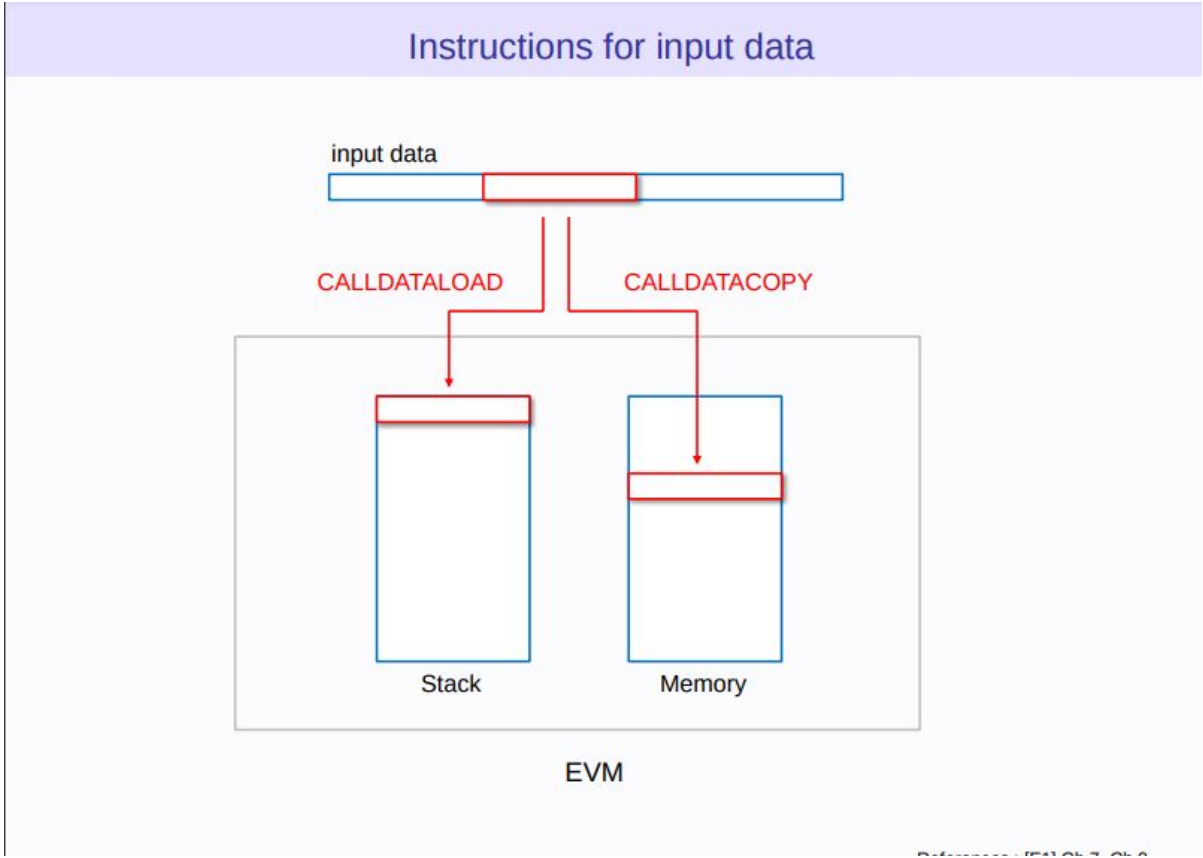
Components of Ethereum Network - Transactions

Input and Output of EVM

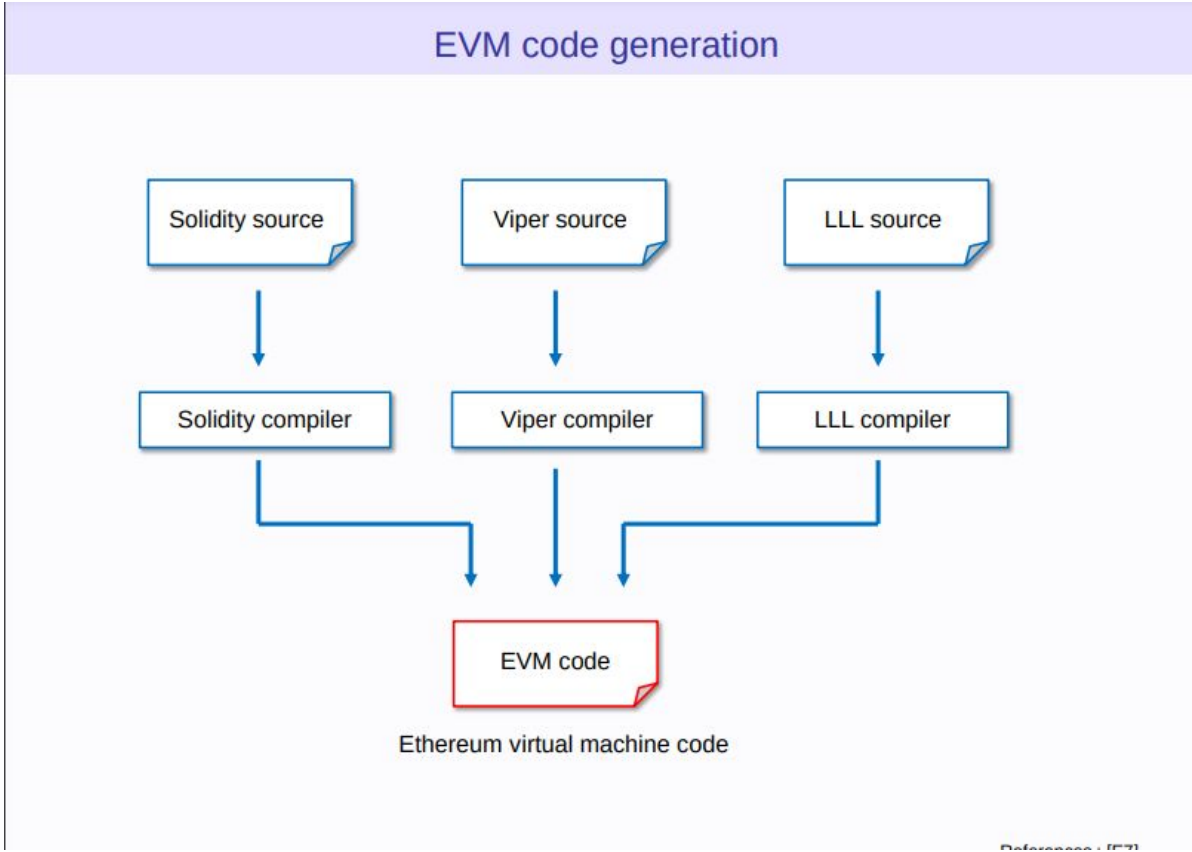


EVM can input external data from a message call.
EVM can output log. EVM can also return values to Caller EVM.

Components of Ethereum Network - Transactions

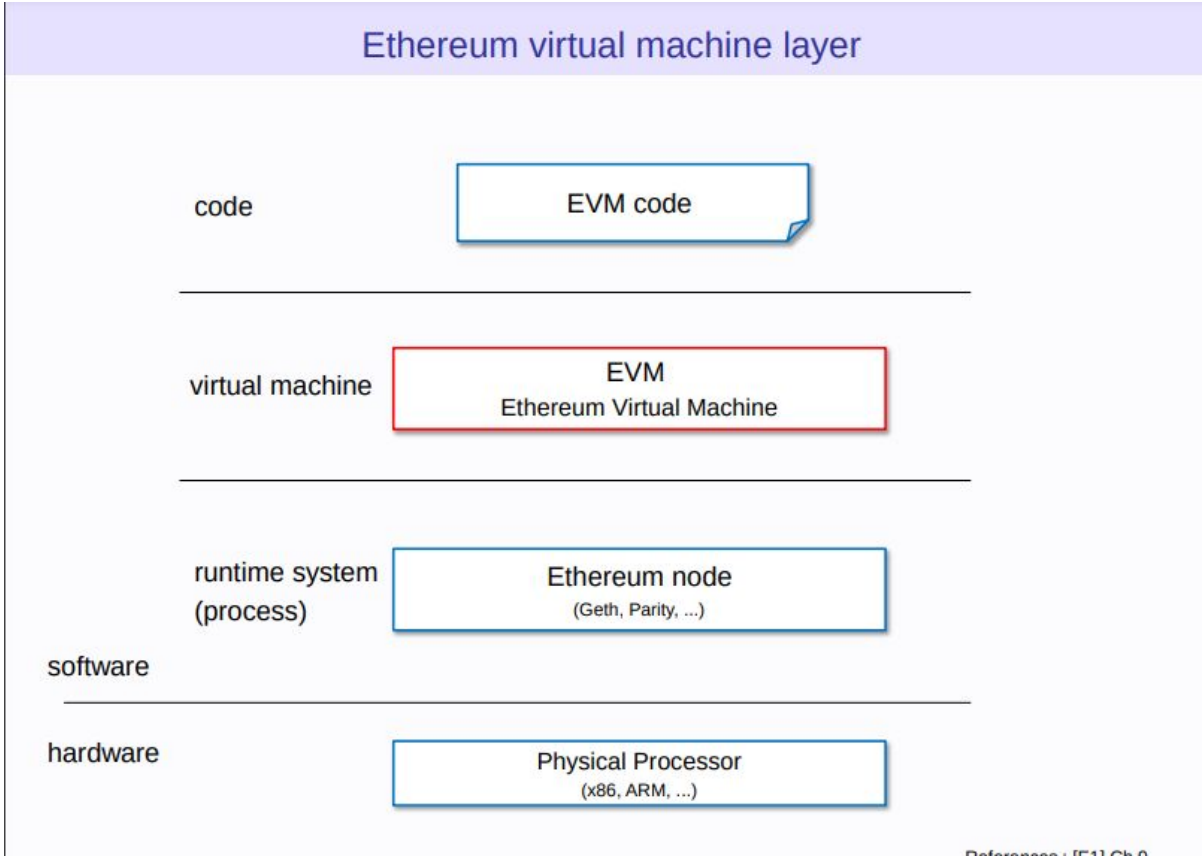


Components of Ethereum Network - EVM



Courtesy : Takenobu T

Components of Ethereum Network - EVM



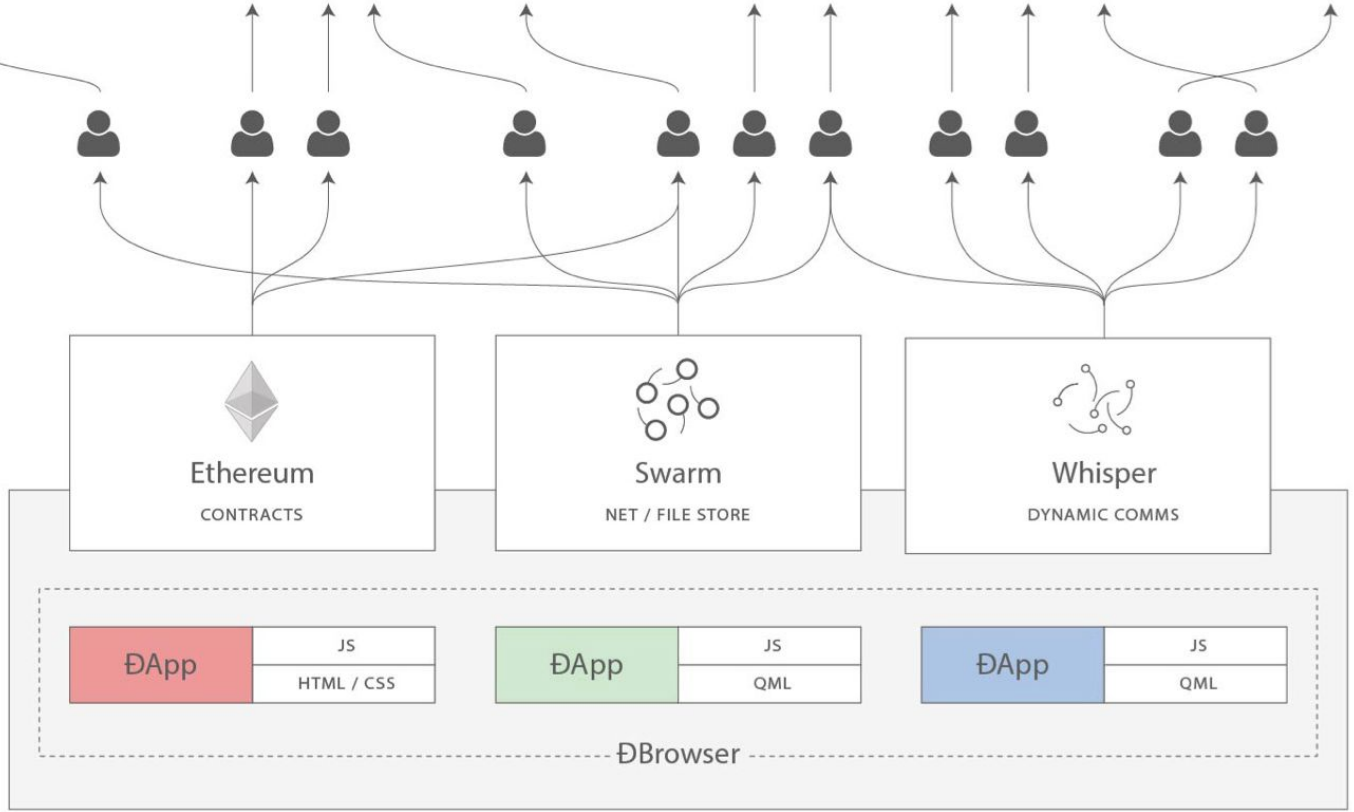
Courtesy : Takenobu T

Components of Ethereum Network - Swarm & Whisper

- Swarm and Whisper are complementary technologies contributing to the vision of Ethereum as a "world computer".
- When imagining Ethereum as a metaphor for a shared computer, it should be noted that computation alone is not enough.
- For a computer to be fully useful, it also needs storage to "remember" things and bandwidth to "communicate" them. This could be summarised as such:
  - **Contracts**: decentralized logic
  - **Swarm**: decentralized storage
  - **Whisper**: decentralized messaging

Courtesy : Ethereum Stackexchange

Components of Ethereum Network - Swarm & Whisper



Interaction including Ethereum contracts, Swarm storage, Whisper comms

Courtesy : Ethereum Stackexchange

Components of Ethereum Network - Swarm & Whisper

Swarm

- designed as an *accounting protocol* that benefits from the automatic execution of so-called "smart contracts" running on the Ethereum Virtual Machine (EVM).
- This accounting protocol is *independent of the physical storage mechanism*.
- That is, it is **not intrinsically tied to a specific storage system**.
  - It could be IPFS, BitTorrent, or some future technology not yet invented.
- it is part of the vision of a fully decentralized web.

Courtesy : Ethereum Stackexchange
Oreilly - Whisper
Etehreum Blog

Components of Ethereum Network - Swarm & Whisper

Whisper

- provides **decentralized peer-to-peer messaging capabilities** to the Ethereum network.
- It is an **identity based messaging system**
- It is a **communication protocol that DApps use to communicate with each other.**
- The data and routing of messages are **encrypted within Whisper communications**.
- uses the DEVp2p wire protocol for exchanging messages between nodes on the network.
- designed to be **used for smaller data transfers** and in scenarios where **real-time communication is not required.**
- **designed to provide a communication layer that cannot be traced**
- provides dark communication between parties.

Note : Blockchain can be used for communication, but that is expensive, and a consensus is not really required for messages exchanged between nodes.

- used as a **protocol that allows censor-resistant communication**.

Courtesy : Ethereum Stackexchange
Oreilly - Whisper
Etehreum Blog

Components of Ethereum Network - Swarm & Whisper

Whisper

- At a considerable cost of bandwidth and latency, whisper ⇒ deliver a 100% dark operation.
- that is **zero leakage of metadata during peer-to-peer communication**
- Normal communication protocol's ⇒ to maximize the bandwidth and minimize latency.
- Goal: **to nullify leakage of metadata and achieve true darkness, where no third party can eavesdrop while two peers are communicating**.
    - For this, whisper is **willing to give up on both bandwidth and latency constraints.**
- Whisper messages are ephemeral (short lived) and have an associated time to live (TTL)

Courtesy : Ethereum Stackexchange
Oreilly - Whisper
Etehreum Blog

Components of Ethereum Network - Swarm & Whisper

Whisper

- Allows nodes in the network communicate with each other.
- Supports broadcasting, user-to-user, encrypted messages, and so on.
- It's **not designed to transfer bulk data**.
- **deliver secure messaging between peers without writing any information to the blockchain**.
- It was **part of the DevP2P wire protocol** but is **now deprecated**.

Courtesy : Ethereum Stackexchange
Oreilly - Whisper
Etehreum Blog

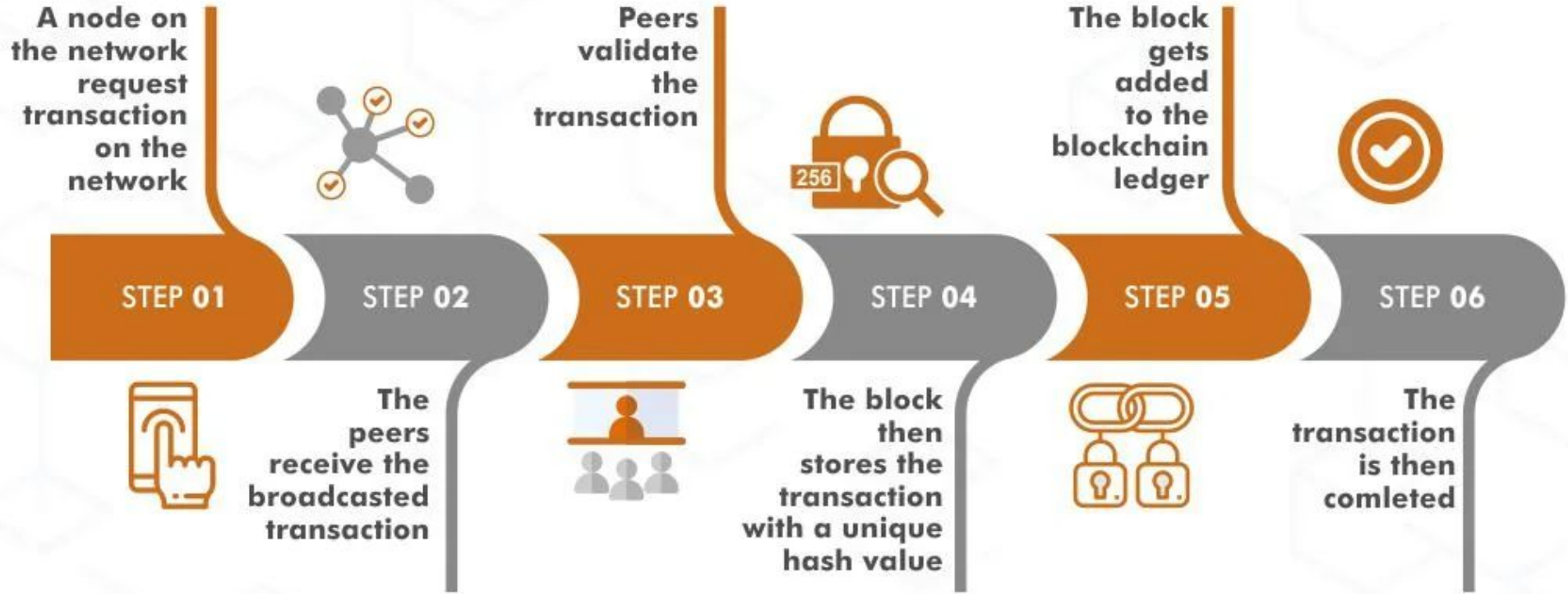
How Does Ethereum Work?

Ethereum **implements an execution environment** called **Ethereum Virtual Machine (EVM).**

1. When a **transaction triggers a smart contract** all the nodes of the network will execute every instruction.
2. **All the nodes** will **run EVM** for **block verification**, where the nodes will go through the transactions listed in the block and runs the code as triggered by the transaction in the EVM.
3. **All the nodes** on the network must **perform the same calculations for keeping their ledgers in sync.**
4. Every transaction must include:
   a. **Gas limit.**
   b. **Transaction Fee** that the sender is willing to pay for the transaction.
5. If **total amount of gas needed to process the transaction < = the gas limit** then the *transaction will be processed*
6. if the **total amount of the gas > the gas limit** then the *transaction will not be processed* *and* the fees are still lost.

Thus it is **safe to send transactions with the gas limit above the estimate to increase the chances of getting it processed**

Transactions in Blockchain - Life Cycle



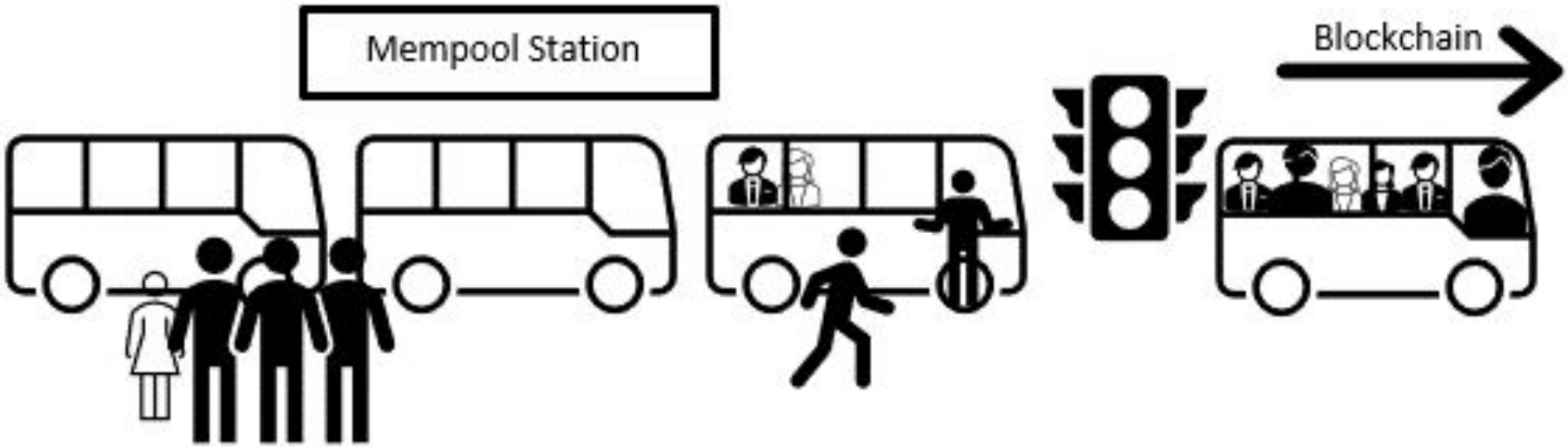
Courtesy : CBCAmerica

Transactions in Blockchain - Life Cycle

1. Someone requests a transaction. The transaction could involve cryptocurrency, contracts, records, or other information.
2. **Transaction is broadcast to all P2P** participation computers in the specific blockchain network. These are called **Nodes**. All transactions are published to the **Mempool** or memory pool, where they are considered 'pending'. **Gas fees** are paid by users as part of the transaction to compensate for the computing energy required to process and validate transactions on the blockchain.
3. **Miners** verify the transaction. Every computer in the network checks the transaction against some validation rules that are set by the creators of the specific blockchain network.
4. **Validated transactions** are stored into a block and are sealed with a lock referred to as the **Hash**.
5. **New block is added to the existing Blockchain**. This block becomes part of the blockchain when other computers in the network validate if the lock on the block is correct.
6. The transaction is complete. Now the transaction is part of the blockchain and cannot be altered in any way.

Courtesy : Ledger.com

Transactions in Blockchain - The Bus Station Analogy

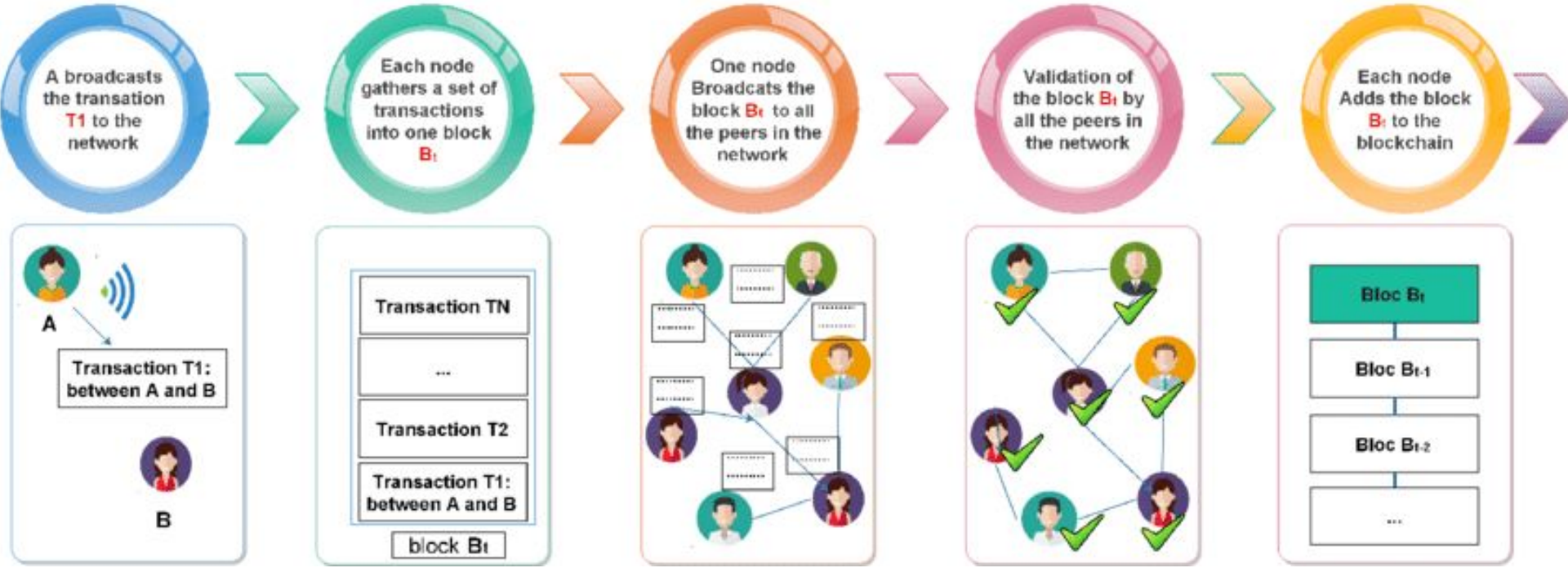


Courtesy : TxStreet

Transactions in Blockchain - Live Demo

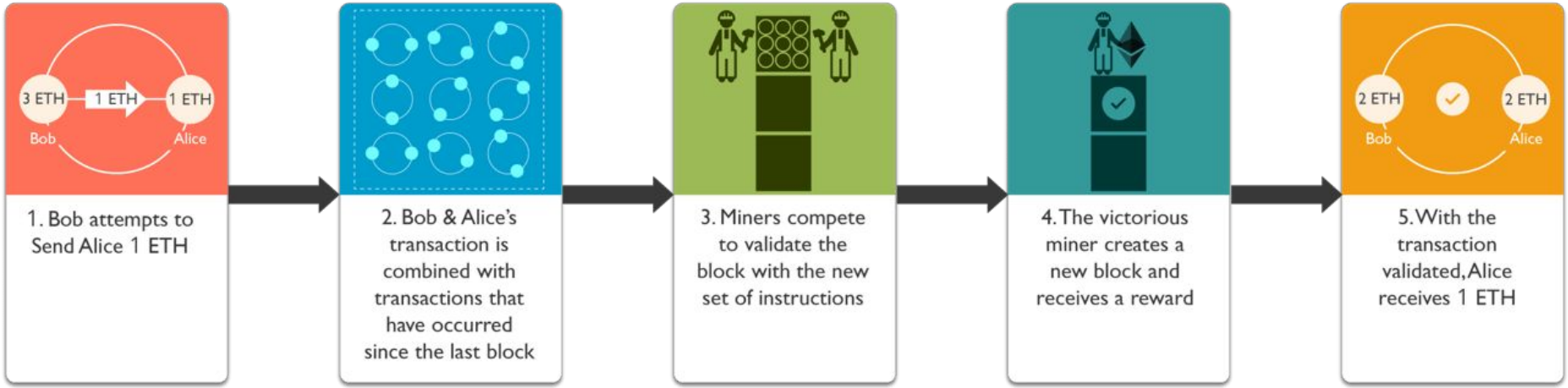
Courtesy : TxStreet

Transactions in Blockchain - Example



Courtesy : Research Gate

Transactions in Blockchain - Example



The proof-of-work algorithm used is called **Ethash**, which is a hashing algorithm inspired by the **Dagger-Hashimoto Algorithm.**

Now that we've seen the working architecture of ethereum and discussed it's essential elements, let's see a real-world problem and the ethereum approach to solve the same.

Courtesy : Edureka

Transactions in Blockchain - Example

- Alice wants to send two coins to Bob.
- Each transaction has **three main parts**:
  - **The input:** Alice's private coin address, she wants to spend.
  - **The output:** Bob's public key or coin address.
  - **Amounts:** the amount of coins Alice wants to spend.

1. **Alice signs a message with the transaction details using her private key.** The message contains the input, output, and amount to be sent.
2. The **transaction is then broadcast to the network** saying the amount of coins in her account should go down by two. The amount in Bob's account should increase by two.
3. **Each computer in the network will receive the message** and apply the requested transaction to its copy of the ledger, updating the account balances.
4. **Add the transactions into the MemPool.**
5. **Miner** select few transactions from MemPool and tries to **solve Cryptographic Puzzle.**
6. On solving the Puzzle, the **Block is broadcasted to the network for validation**.
7. After adding Block into the Blockchain, **Transactions added in the Block are later removed from MemPool**

Courtesy : Research Gate

End-End Transaction in Ethereum

Sam wants to send $$ to Mark

Transaction reaches the network

The nodes compete and validate the transaction

A new block with transaction is generated

Both Accounts are updated

The Block is replicated and stored on all nodes

Courtesy : Packt Pub

Information in a Submitted Transaction in Ethereum

- from – the address of the sender, that will be signing the transaction. This will be an externally-owned account as contract accounts cannot send transactions.
- recipient – the receiving address (if an externally-owned account, the transaction will transfer value. If a contract account, the transaction will execute the contract code)
- signature – the identifier of the sender. This is generated when the sender's private key signs the transaction and confirms the sender has authorized this transaction
- nonce - a sequentially incrementing counter which indicates the transaction number from the account
- value – amount of ETH to transfer from sender to recipient (denominated in WEI, where 1ETH equals 1e+18wei)
- input data – optional field to include arbitrary data
- gasLimit – the maximum amount of gas units that can be consumed by the transaction. The EVM specifies the units of gas required by each computational step
- maxPriorityFeePerGas - the maximum price of the consumed gas to be included as a tip to the validator
- maxFeePerGas - the maximum fee per unit of gas willing to be paid for the transaction (inclusive of baseFeePerGas and maxPriorityFeePerGas)

Courtesy : Packt Pub

Ethereum Vs Bitcoin

Basis	Bitcoin	Ethereum
Smart Contracts	Although bitcoin do have smart contracts, they are not as flexible or complete as Ethereum smart contracts. Smart contracts in Bitcoin does not have all the functionality that a programming language would give them.	Ethereum allows us to create smart contracts. Smart contracts are computer codes that is stored on a blockchain and executed when the predetermined terms and conditions are met.
Smart Contract Programming Language	Smart contracts on Bitcoin are written in programming languages like Script, Clarity.	Smart contracts on Ethereum are written in programming languages like Solidity, Vyper, etc.
Transactions	Generally, bitcoin transactions are only for keeping notes.	Ethereum transactions may contain some executable code.

Ethereum Vs Bitcoin

Basis	Bitcoin	Ethereum
Definition	Bitcoin (abbreviation: BTC; sign: ₿) is a decentralized digital currency that can be transferred on the peer-to-peer bitcoin network.	Ethereum is a decentralized global software platform powered by blockchain technology. It is most commonly known for its native cryptocurrency, ether (ETH).
History	The word bitcoin was defined in a white paper published on 31 October 2008. The currency began use in 2009.	Ethereum was conceived in 2013 by programmer Vitalik Buterin, and then went live on 30 July 2015.
Purpose	The purpose of bitcoin was to replace national currencies during the financial crisis of 2008.	The purpose of Ethereum was to utilize blockchain technology for maintaining a decentralized payment network and storing computer code.

Ethereum Vs Bitcoin

Basis	Bitcoin	Ethereum
Hash Algorithm	Bitcoin runs on the **SHA-256** hash algorithm.	Ethereum runs on the **Keccak-256** hash algorithm.
Consensus Mechanism	The Proof-of-Work (PoW) is the consensus mechanism used by the Bitcoin network.	The Proof-of-Stake is the consensus mechanism used by Ethereum.
Block Time	The block time of bitcoin is 10 minutes.	The block time of Ethereum is 14 to 15 seconds.
Block Limit	The bitcoin blockchain has a block limit of 1 MB.	The Ethereum blockchain does not have a block limit.

Ethereum Vs Bitcoin

Basis	Bitcoin	Ethereum
Popularity	Bitcoin is the most popular digital currency in the market to date.	Ether, native currency of Ethereum is the second-largest cryptocurrency after bitcoin to date.
Energy Consumption	Energy consumption is very high.	Energy consumption is very low as compared to bitcoin
Energy Consumption rate	Energy consumption rate of bitcoin mining system 3.2 Million household.	Energy consumption rate of bitcoin mining system 1.2 Million household.
Structure	Structure of bitcoin is simple and robust.	Structure of Ethereum is complex and feature rich

Ethereum Vs Bitcoin

Basis	Bitcoin	Ethereum
Rewards	Miner got nearly 6.25 BTC on successfully adding new block in network.	Miner got nearly 5 BTC along with same additional rewards on successfully adding new block in network.
Assets	Assets of Bitcoin is BTC.	Assets of Ethereum is Ether.

Types of test-networks used in Ethereum

Ethereum networks

- groups of connected computers that communicate using the Ethereum protocol.
- There is **only one Ethereum Mainnet,**
- But **independent networks**
  - conforms to the **same protocol rules** can be created
  - for testing and development purposes.
  - These independent "networks" that conform to the protocol **without interacting with each other.**

Note :

- One can even start one locally on your own computer for testing your smart contracts and web3 apps.
- Eg : Geth & Ganache Private Networks created during the Lab Experiments
- Yout **Ethereum account will work across the different networks**,
- But your account balance and transaction history won't carry over from the main Ethereum network.

For testing purposes :

- it's useful to know which networks are available
- how to get testnet ETH to play around with.

For security considerations : it's not recommended to reuse mainnet accounts on testnets or vice versa.

Courtesy : Ethereum

Types of test-networks used in Ethereum

- **PUBLIC NETWORKS**
  - Public networks are accessible to anyone in the world with an internet connection.
  - Anyone can read or create transactions on a public blockchain and validate the transactions being executed.
  - The consensus among peers decides on the inclusion of transactions and the state of the network.
- **Ethereum Mainnet**
  - Mainnet is the primary public Ethereum production blockchain,
  - where actual-value transactions occur on the distributed ledger.
  - When people and exchanges discuss ETH prices, they're talking about Mainnet ETH.

Courtesy : Ethereum

Types of test-networks used in Ethereum

- **Ethereum Testnets**
  - 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 Mainnet.
  - Test any contract code you write on a testnet before deploying to Mainnet.
  - Among dapps that integrate with existing smart contracts, most projects have copies deployed to testnets.
  - Most testnets started by using a permissioned proof-of-authority consensus mechanism.
  - This means a small number of nodes are chosen to validate transactions and create new blocks – staking their identity in the process.
  - Alternatively, some testnets feature an open proof-of-stake consensus mechanism where everyone can test running a validator, just like Ethereum Mainnet.
  - **ETH on testnets is supposed to have no real value**;
  - Most people get testnet ETH for free from faucets.
  - Most faucets are webapps where you can input an address which you request ETH to be sent to.

Types of test-networks used in Ethereum

- **Which Testnet should I use?**
  - The two public testnets that client developers are **Sepolia** and **Goerli**.
  - **Sepolia**
    - recommended default testnet for application development.
    - Features :
      - Closed validator set, controlled by client & testing teams
      - New testnet, less applications deployed than other testnets
      - Fast to sync and running a node requires minimal disk space
  - **Goerli (long-term support)**
    - the Goerli testnet is deprecated and will be replaced by Holesovice in 2023.
    - Features :
      - Open validator set, stakers can test network upgrades
      - Large state, useful for testing complex smart contract interactions
      - Longer to sync and requires more storage to run a node

Courtesy : Ethereum

Types of test-networks used in Ethereum

- **PRIVATE NETWORKS**
  - An Ethereum network is a private network if its nodes are not connected to a public network (i.e. Mainnet or a testnet).
  - Private only means reserved or isolated, rather than protected or secure.
- **Development networks**
  - To develop an Ethereum application, run it on a private network to see how it works before deploying.
  - Create a local blockchain instance to test your dapp.
  - This allows for much faster iteration than a public testnet.
- **Consortium networks**
  - The consensus process is controlled by a pre-defined set of nodes that are trusted.
  - For example :
    - a private network of known academic institutions that each govern a single node,
    - and blocks are validated by a threshold of signatories within the network.

Note :

If a **public Ethereum network ⇒ public internet**, a **consortium network ⇒ private intranet.**

Metamask

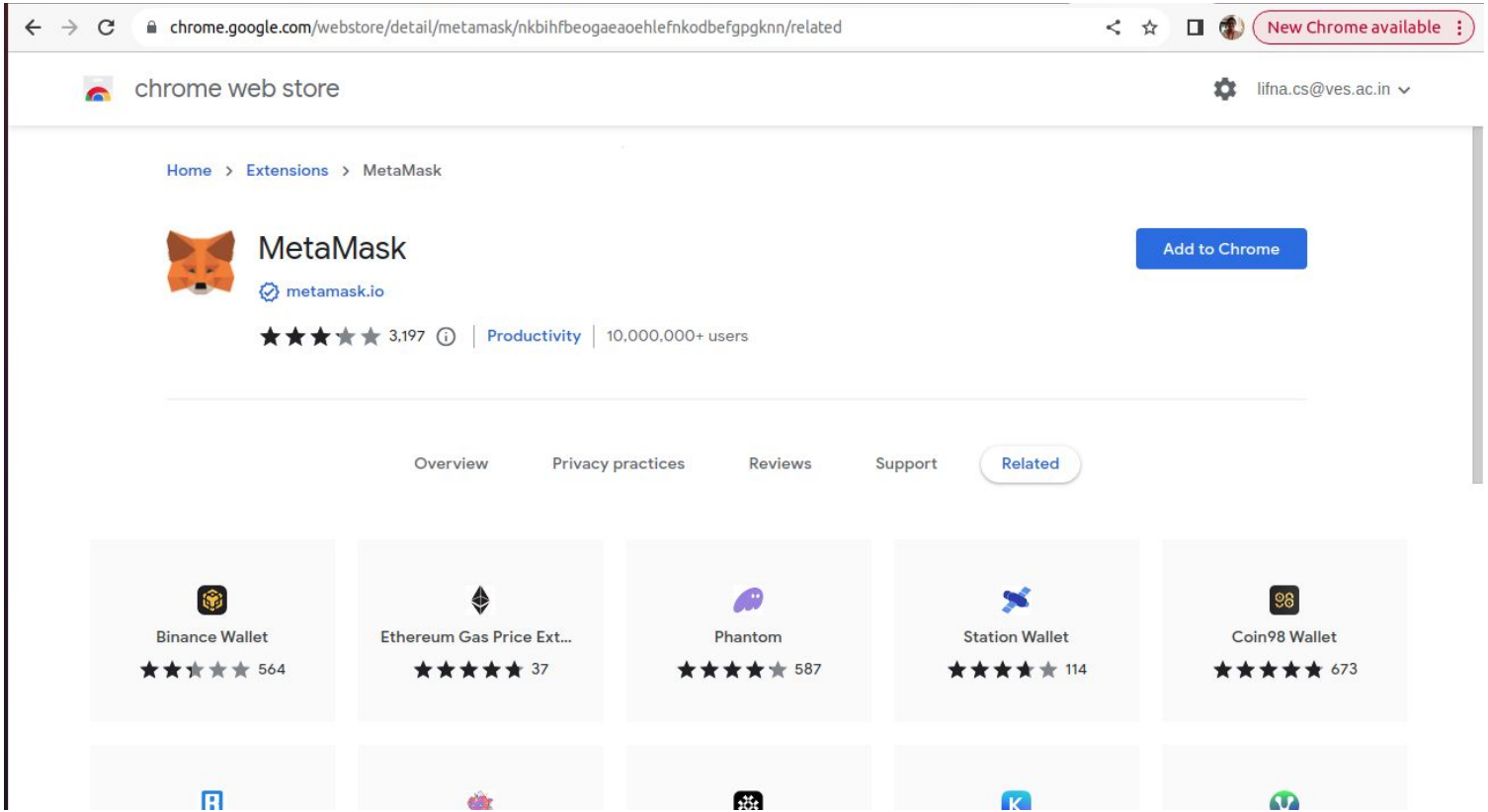
- A software (hot storage) crypto wallet that is mainly used to interact with the Ethereum blockchain.
- One can access MetaMask through a browser extension or mobile app, which is then used to interact with decentralized applications (Dapps).

How to Set Up a MetaMask Wallet

1. Add the chrome extension or download the app
2. Create a wallet
   - Once you choose Get Started, you will be asked to either Import Wallet or Create a Wallet.
3. Create a password
4. Secure your secret phrase
   - make sure you write it down on a piece of paper and store it in a secure location.
   - For more security, write it down on multiple pieces of paper and store it in multiple secure locations.
5. Confirm your secret recovery phrase
   - After writing down your secret phrase you will be asked to enter it again to ensure you wrote it down correctly.
   - Upon entering your phrase, you will be logged into your MetaMask wallet.

Setting up a Metamask Wallet

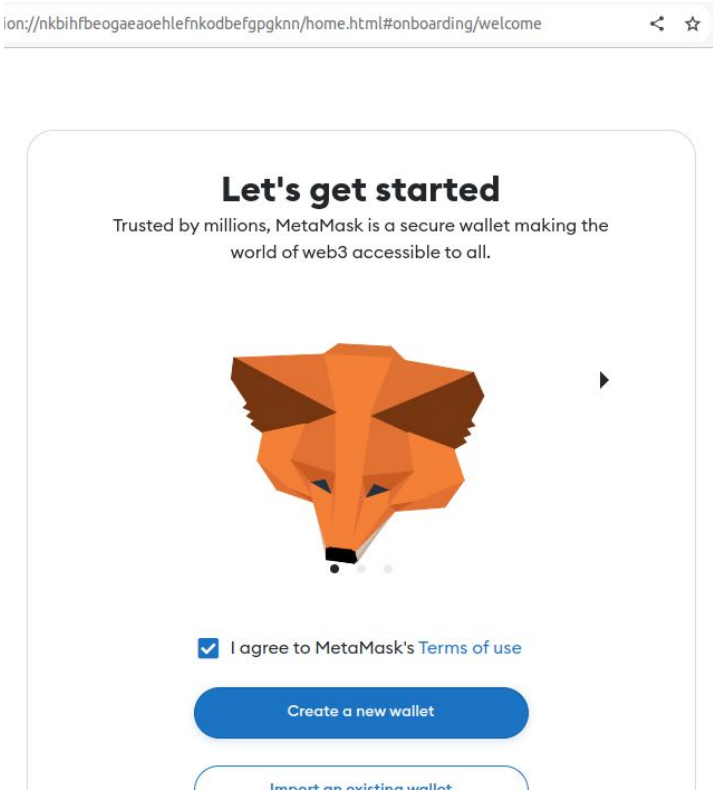
1. Add the Metamask extension

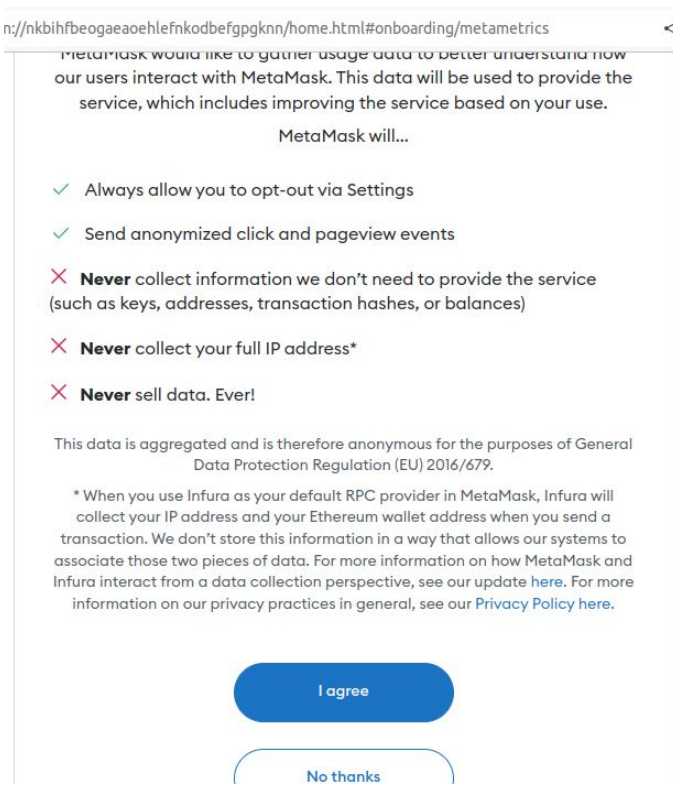


Courtesy : Metamask

Setting up a Metamask Wallet

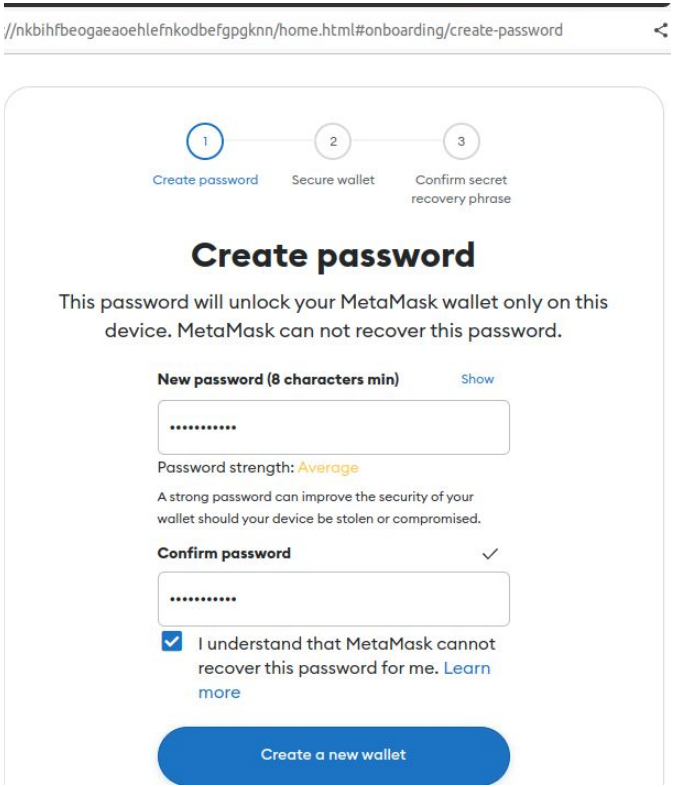
2. Create a wallet





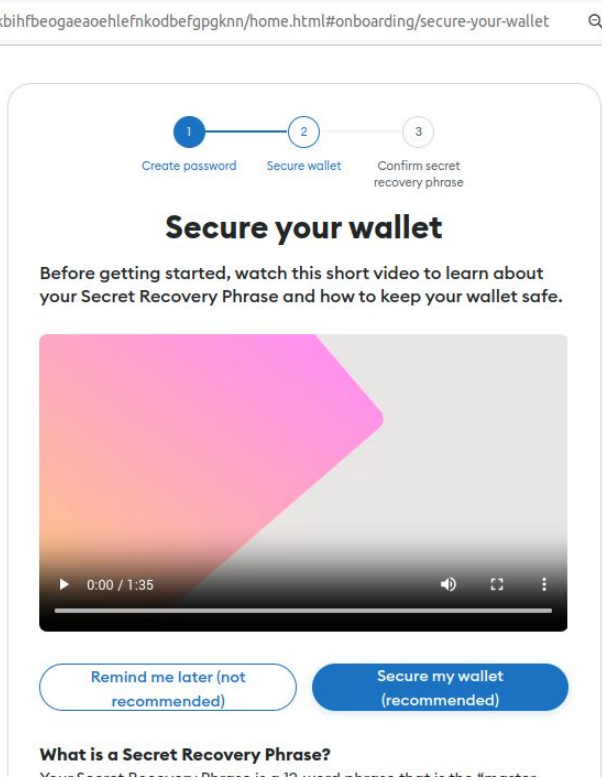
Setting up a Metamask Wallet

3. Create a password

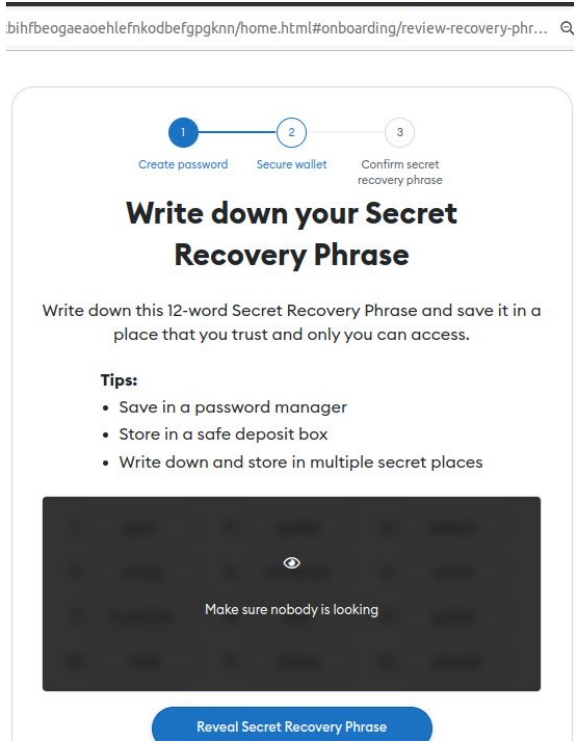


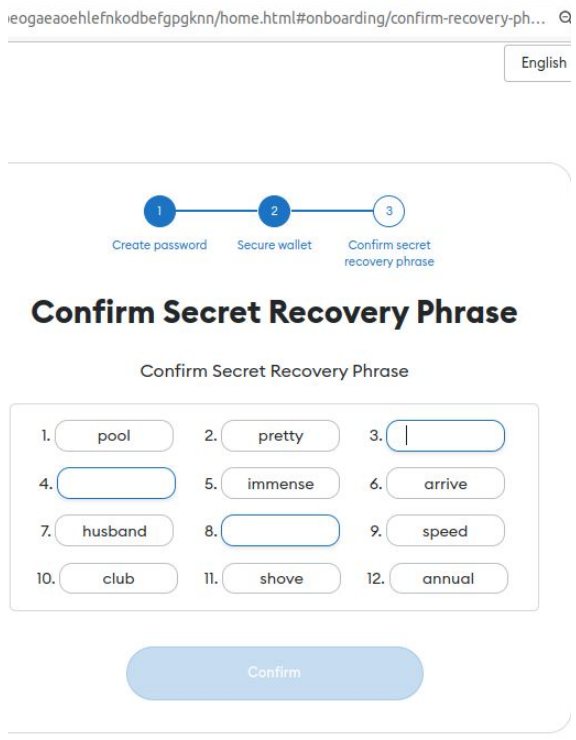
Setting up a Metamask Wallet

4. Secure your secret phrase



5. Confirm your secret recovery phrase

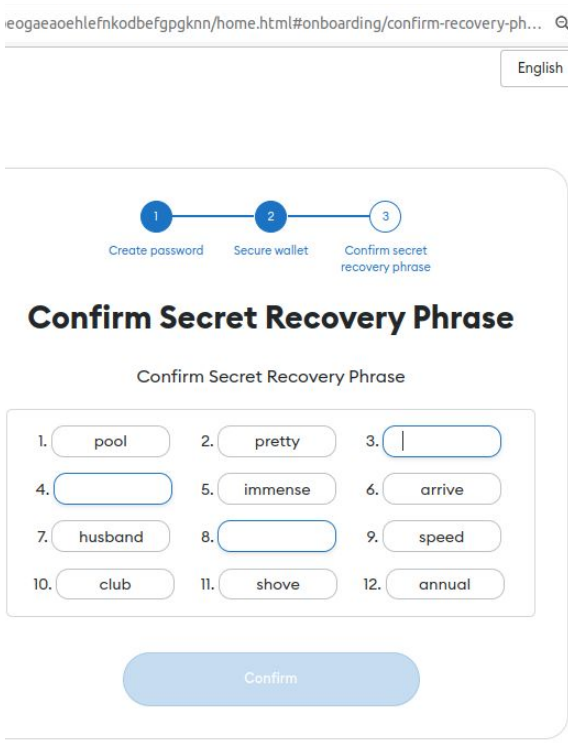


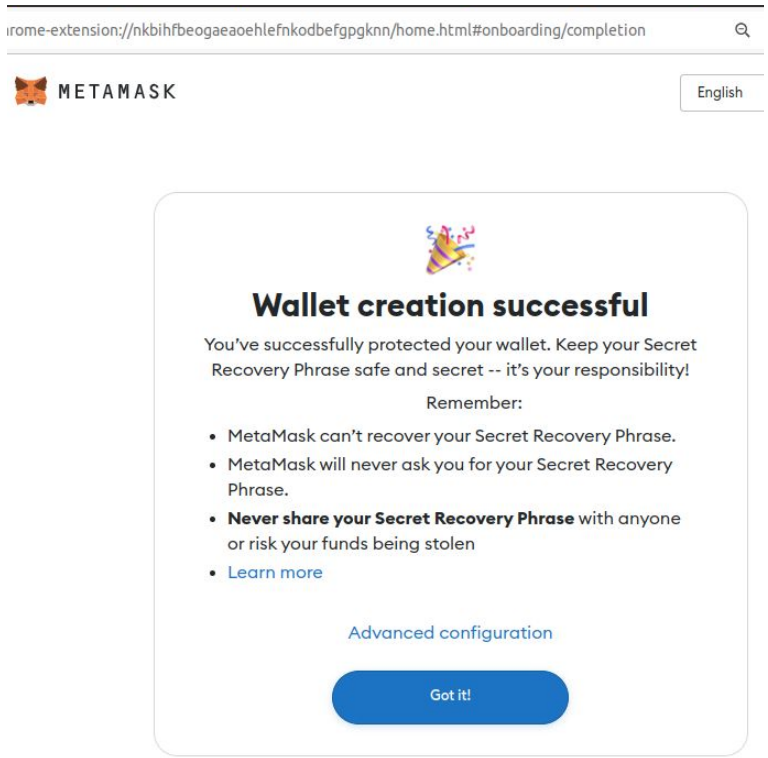


Courtesy : Metamask

Setting up a Metamask Wallet

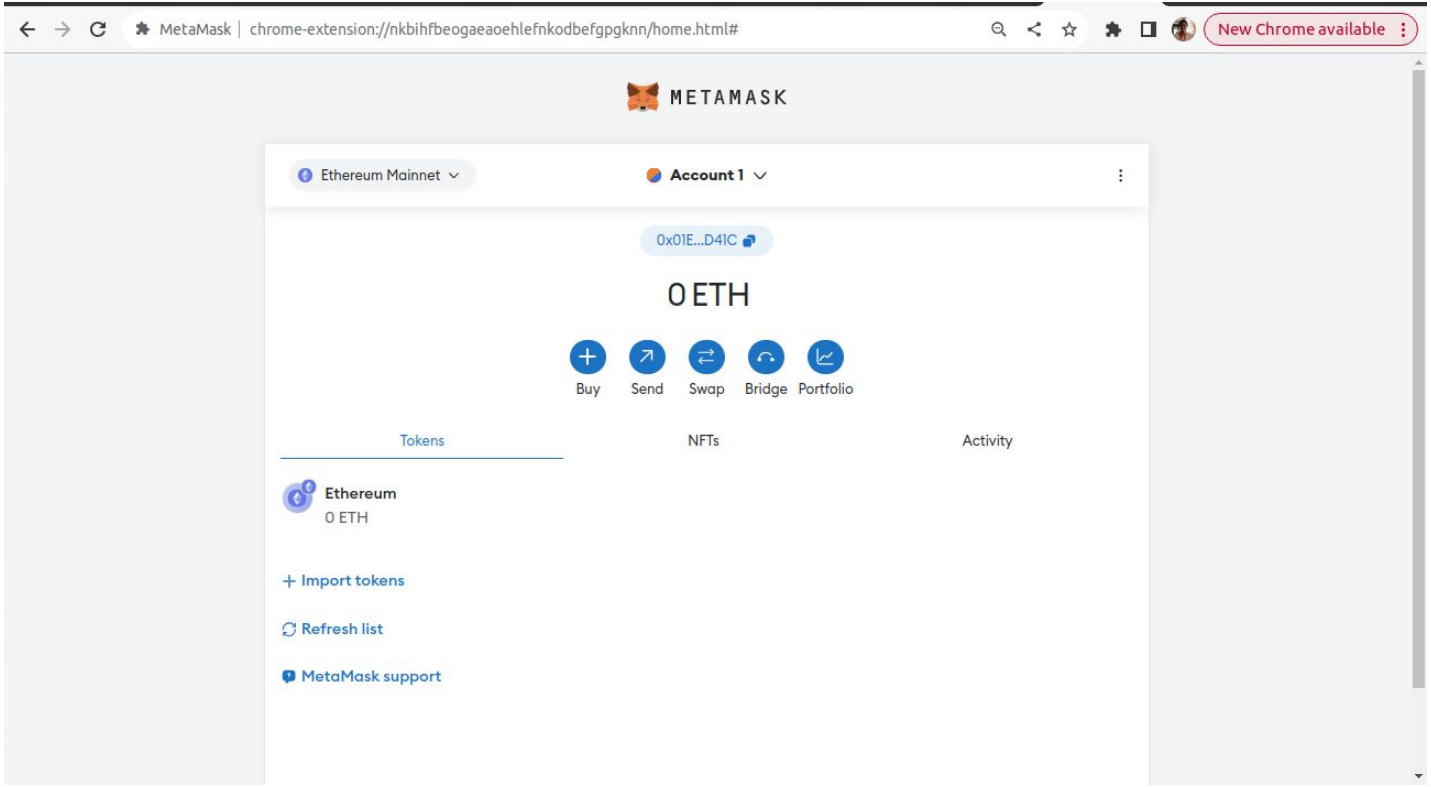
5. Confirm your secret recovery phrase





Transferring Ethers using Metamask

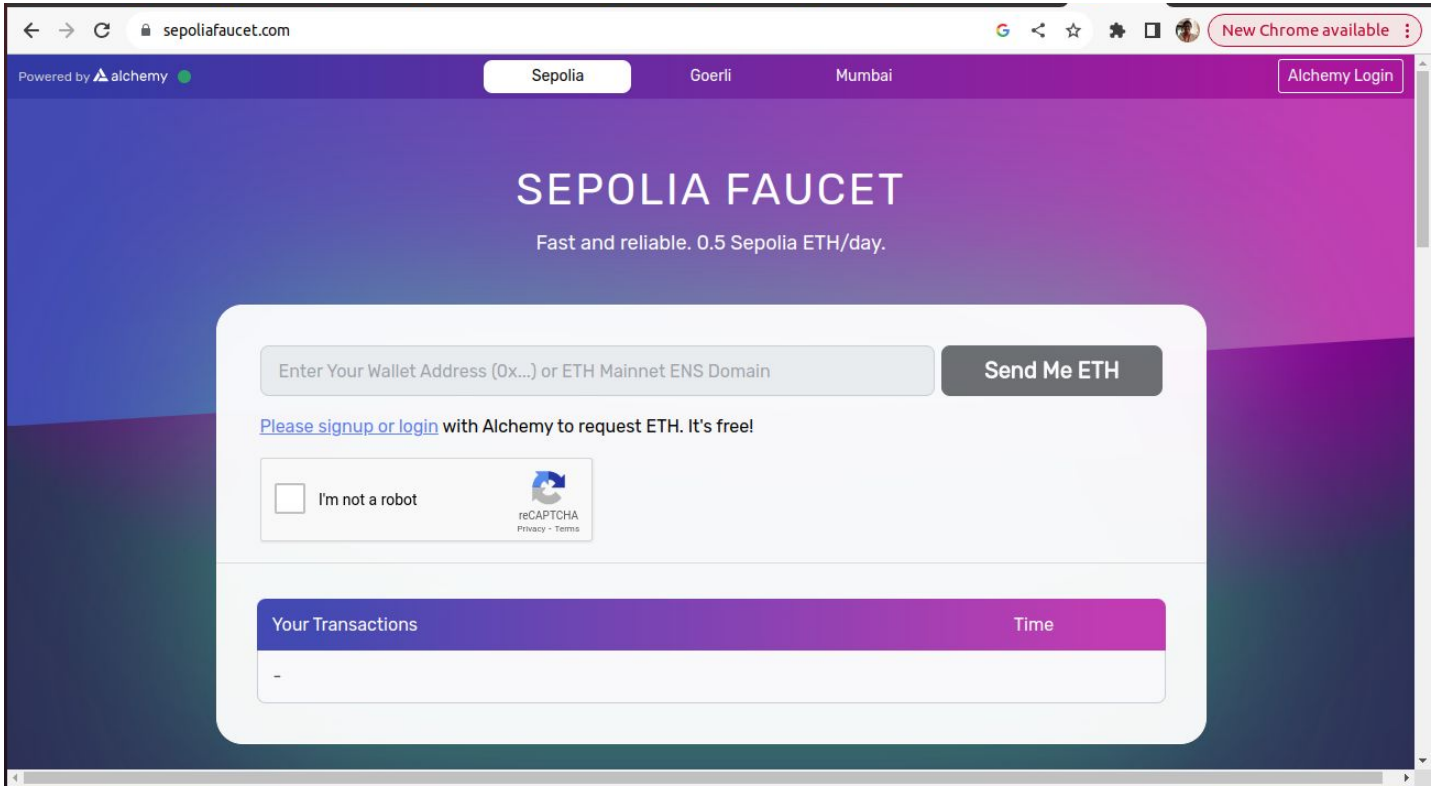
Default Account is created on the Ethereum Mainnet with 0 ETH



Courtesy : Metamask

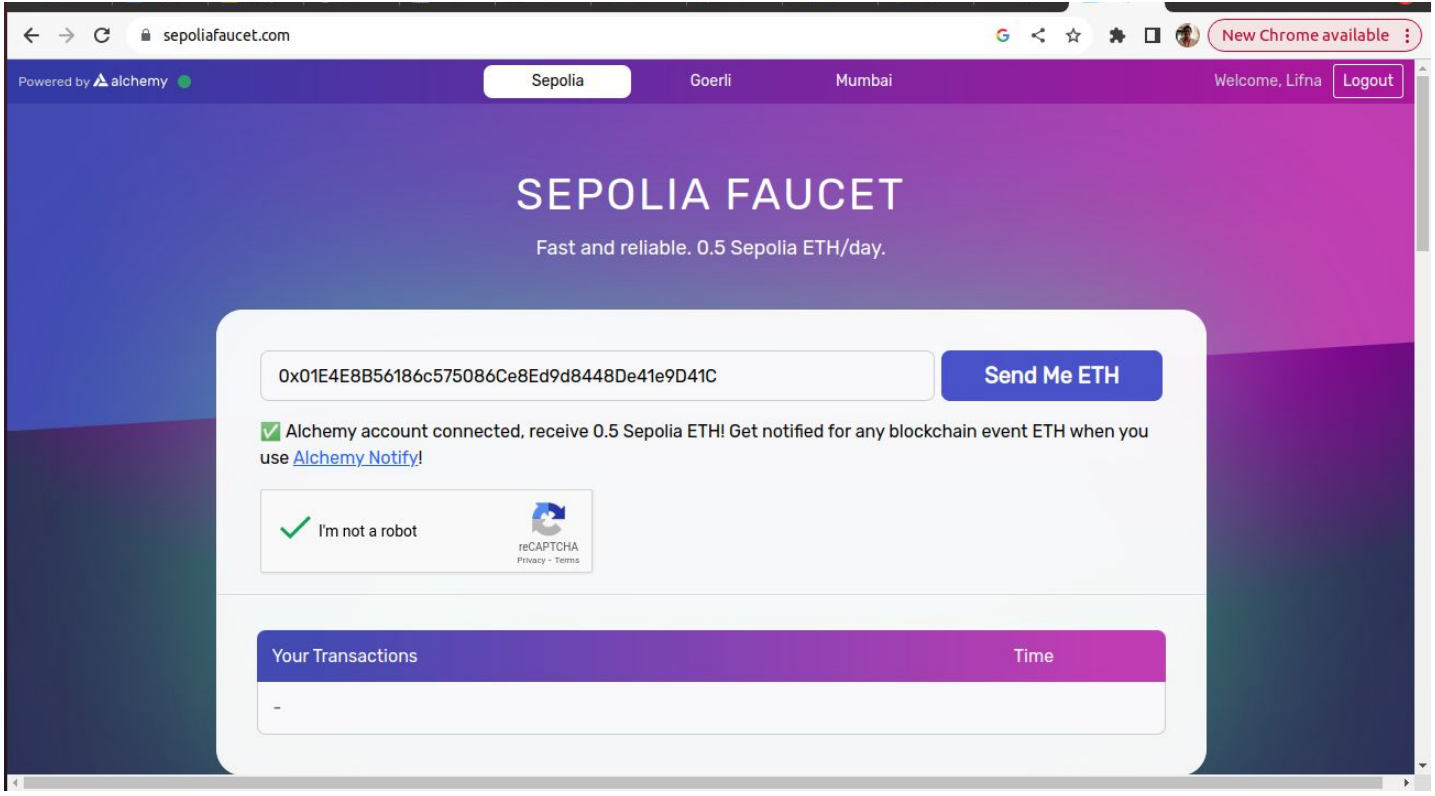
Transferring Ethers using Metamask

Go to sepoliafaucet.com and create an Alchemy Account



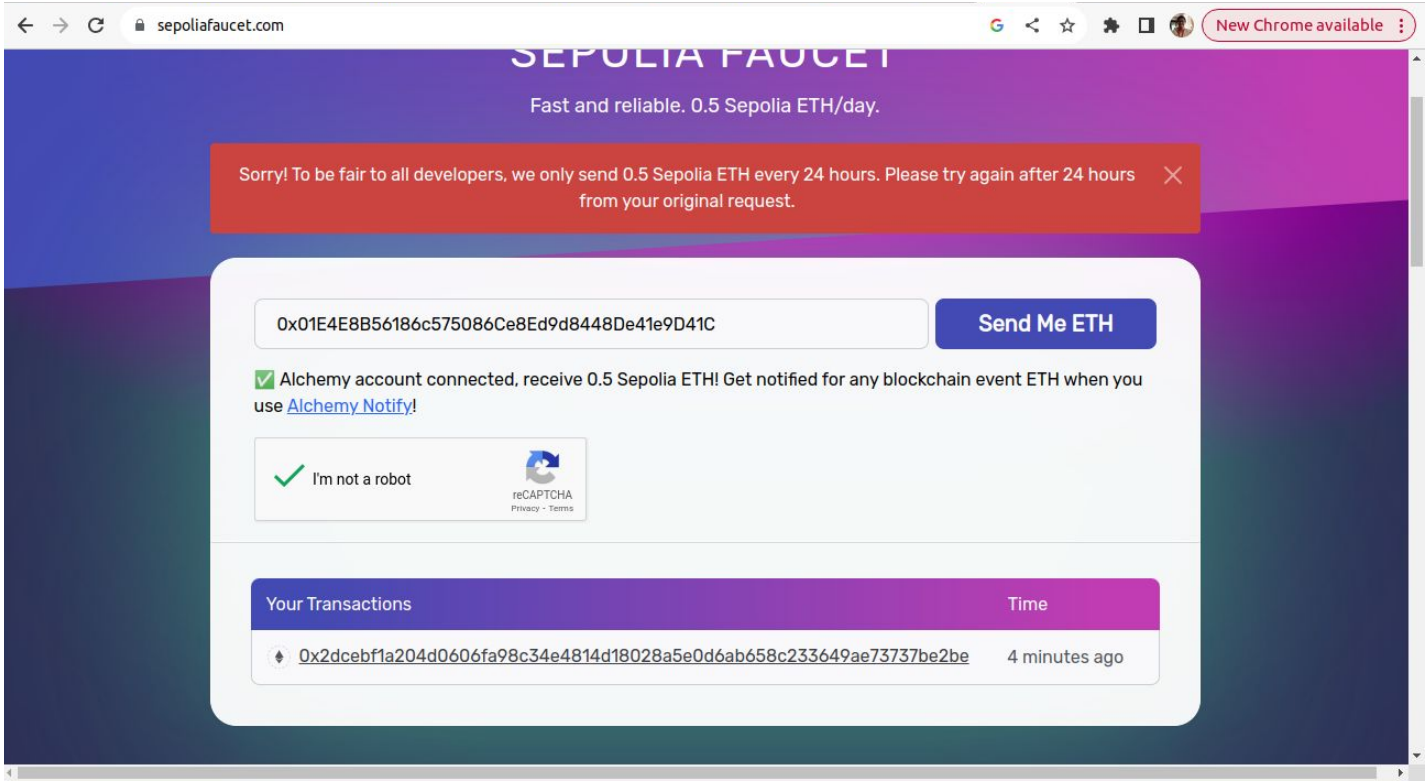
Transferring Ethers using Metamask

Paste the Public Key of the Metamask Account to claim 0.5 ETH/day



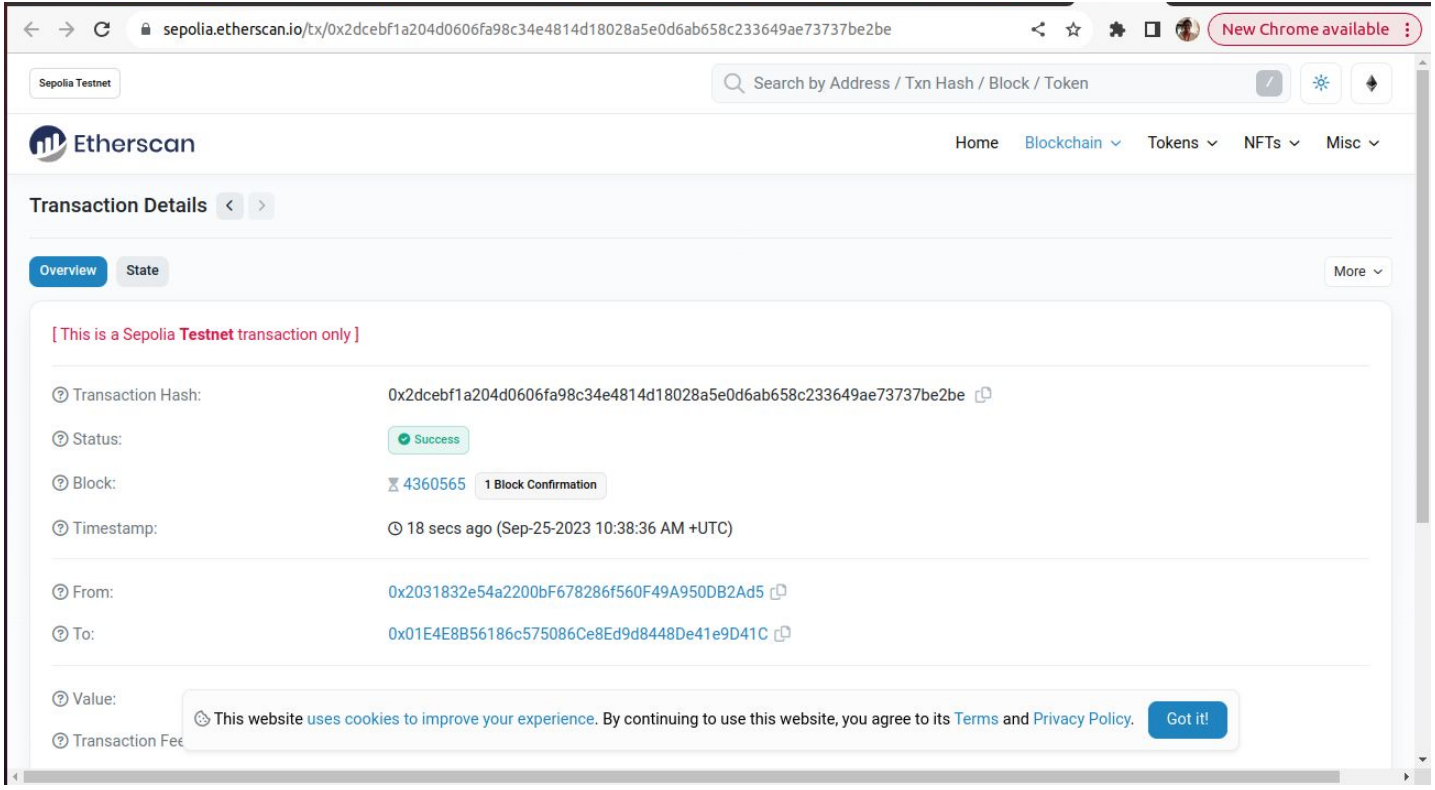
Transferring Ethers using Metamask

Transaction details of the fund transfer is displayed



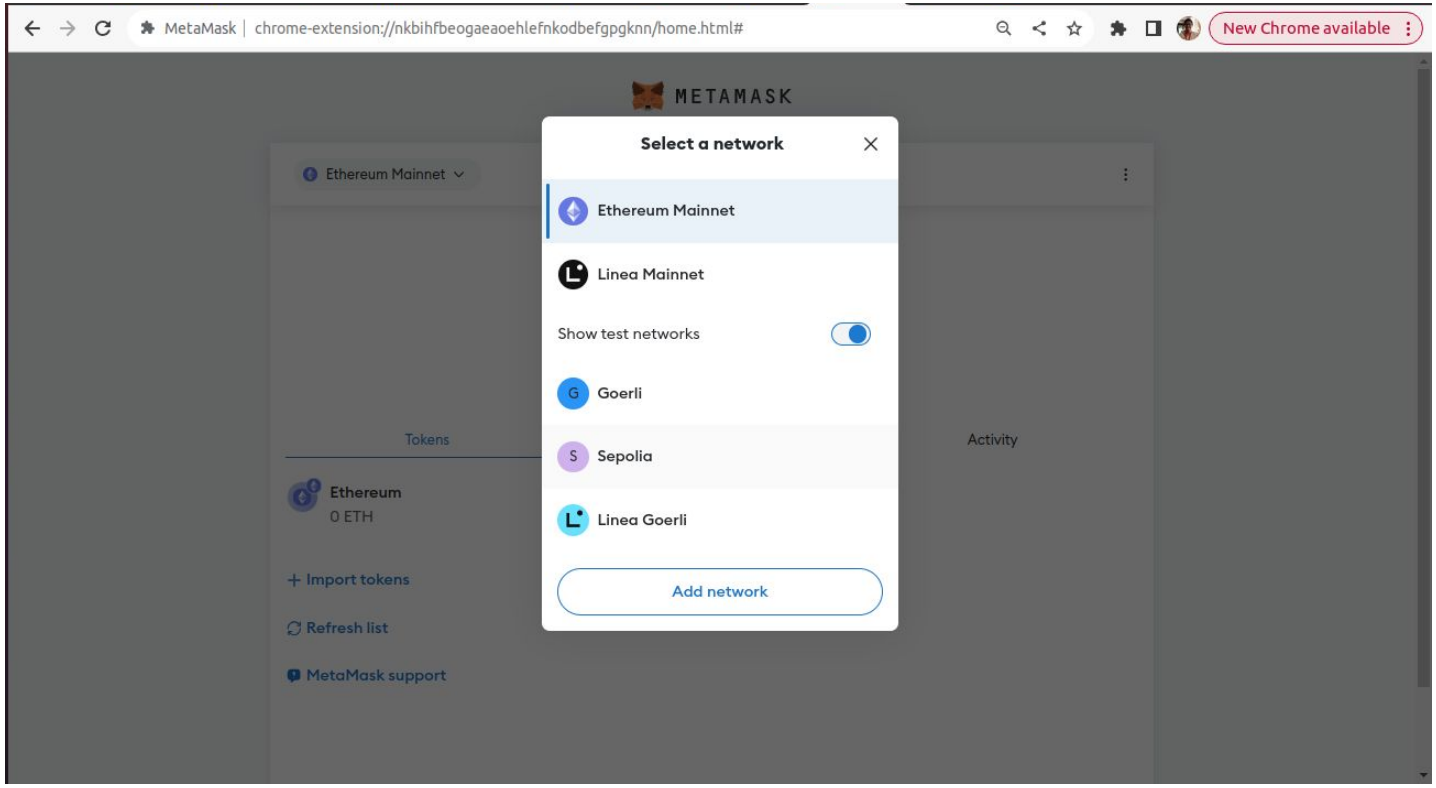
Transferring Ethers using Metamask

On Etherscan.io, the details of the transaction is displayed (@Sepolia Testnet)



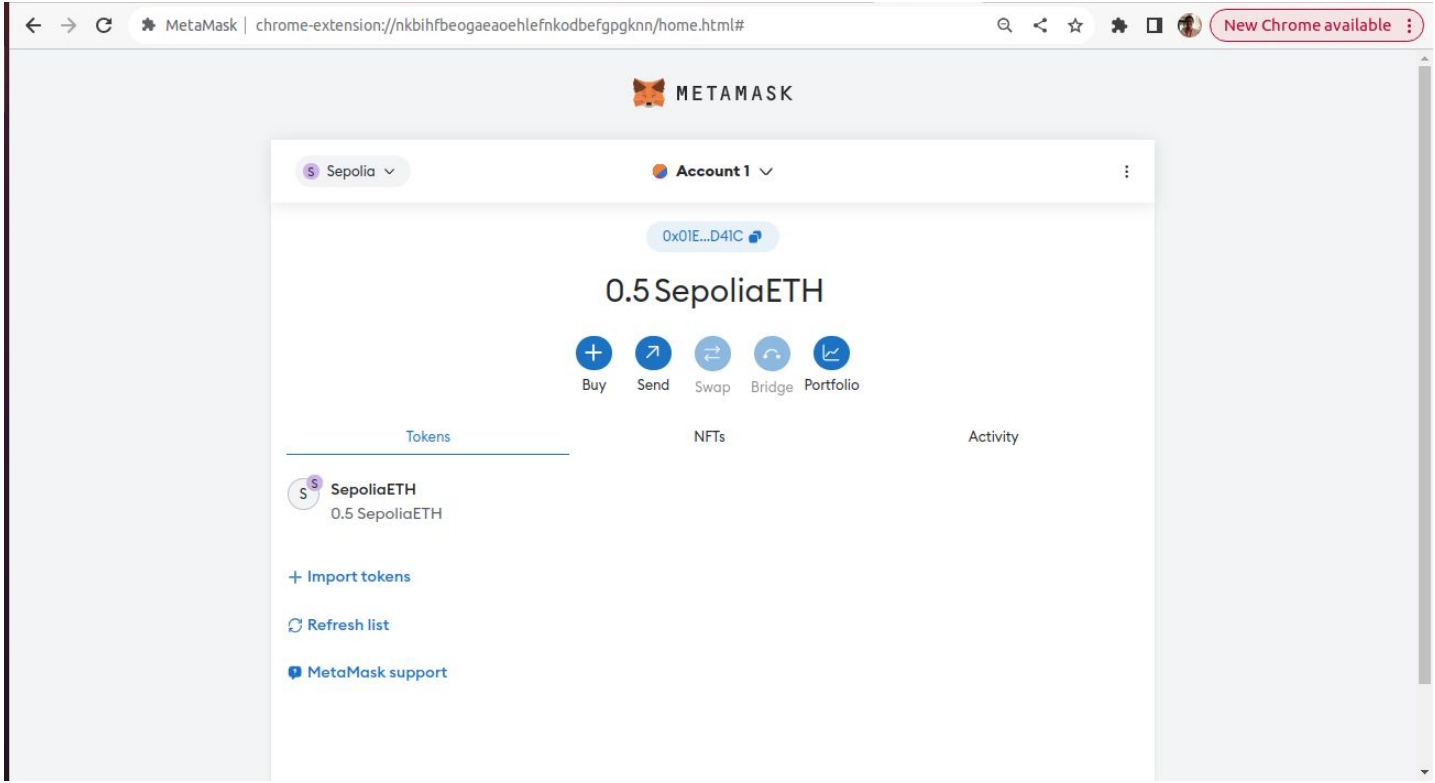
Transferring Ethers using Metamask

On the Metamask, switch to Sepolia Testnet



Transferring Ethers using Metamask

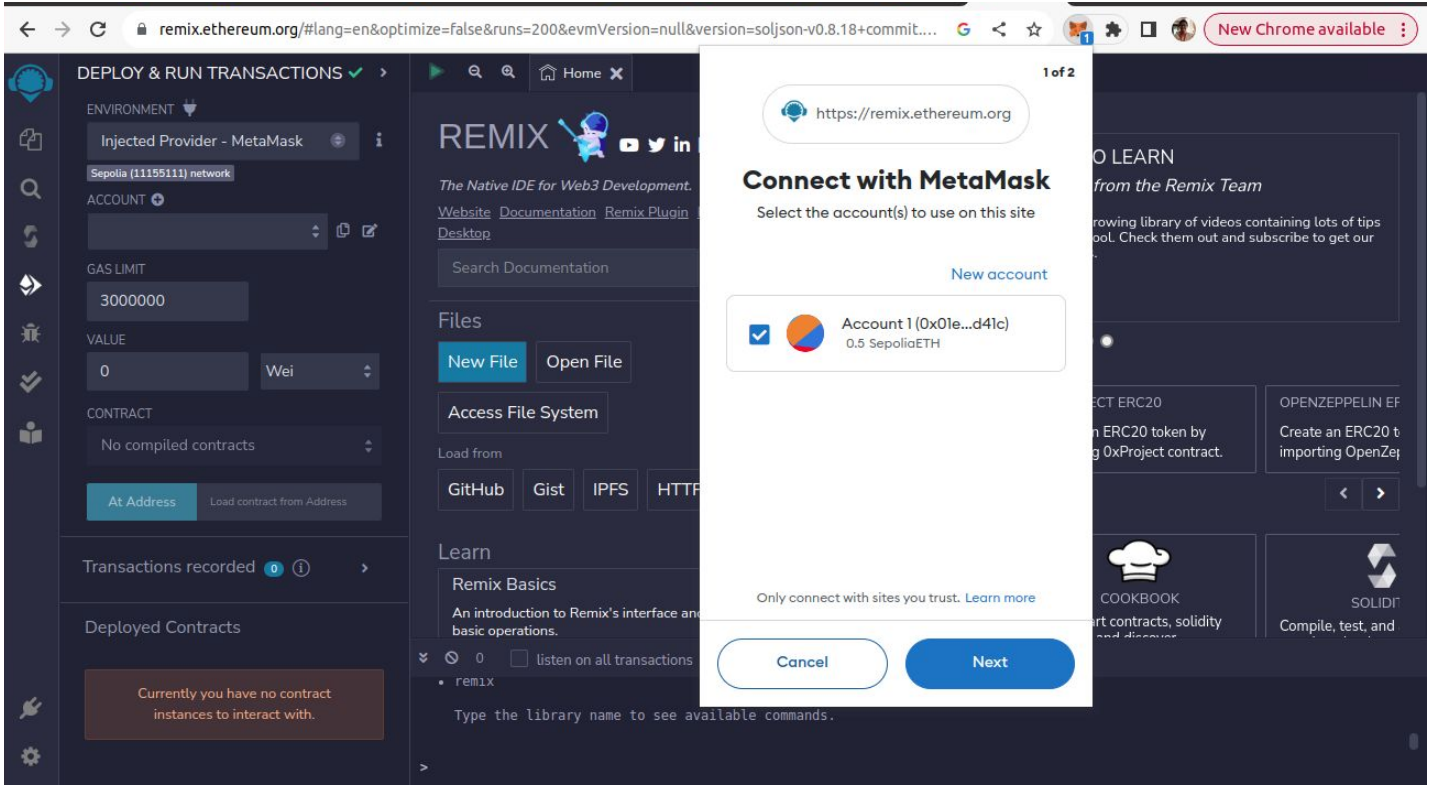
On the Metamask, under the Sepolia Testnet, the Account got a credit of 0.5ETH is updated



Courtesy : Metamask

Transferring Ethers using Metamask

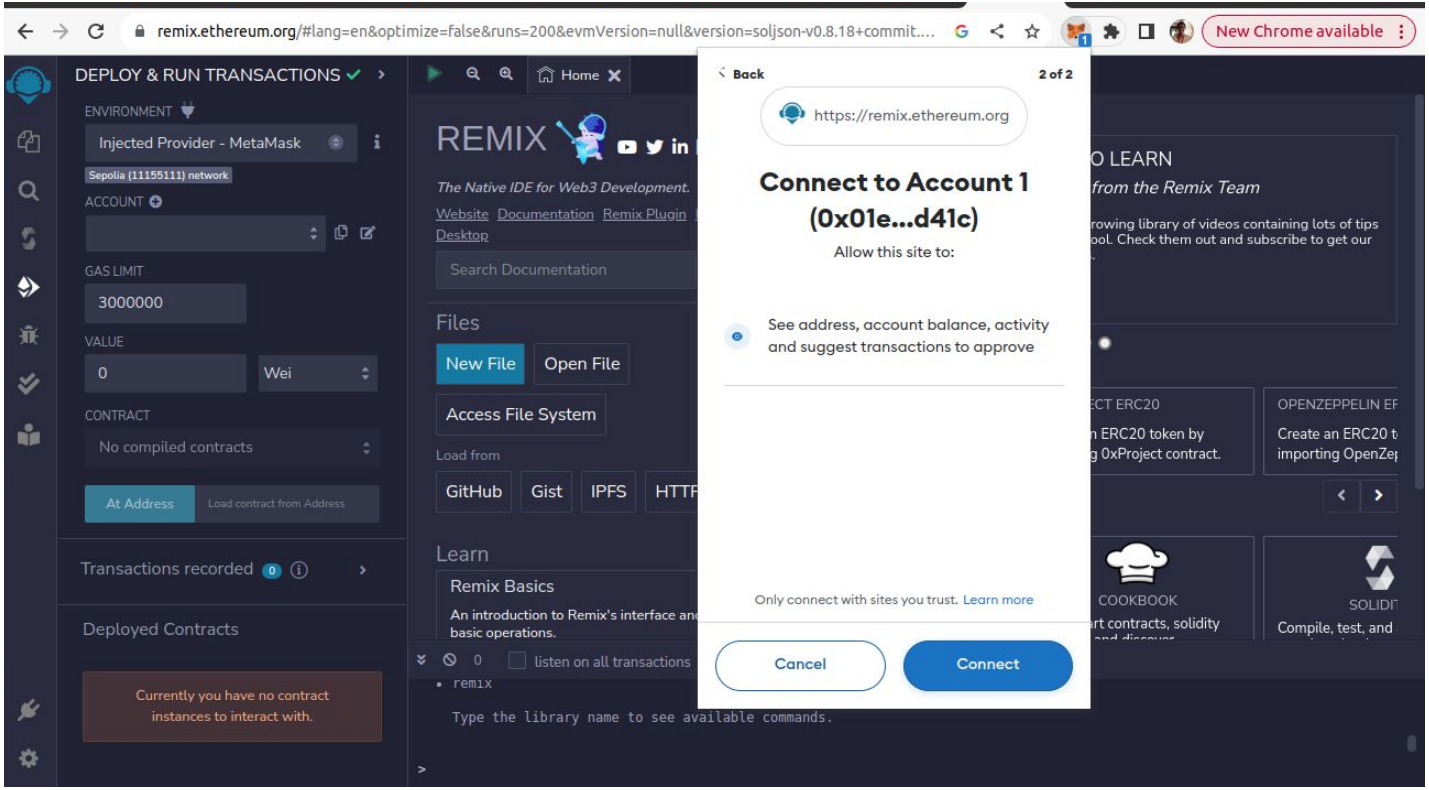
On the Remix IDE, select the Environment as Injected Provider - Metamask



Courtesy : Metamask

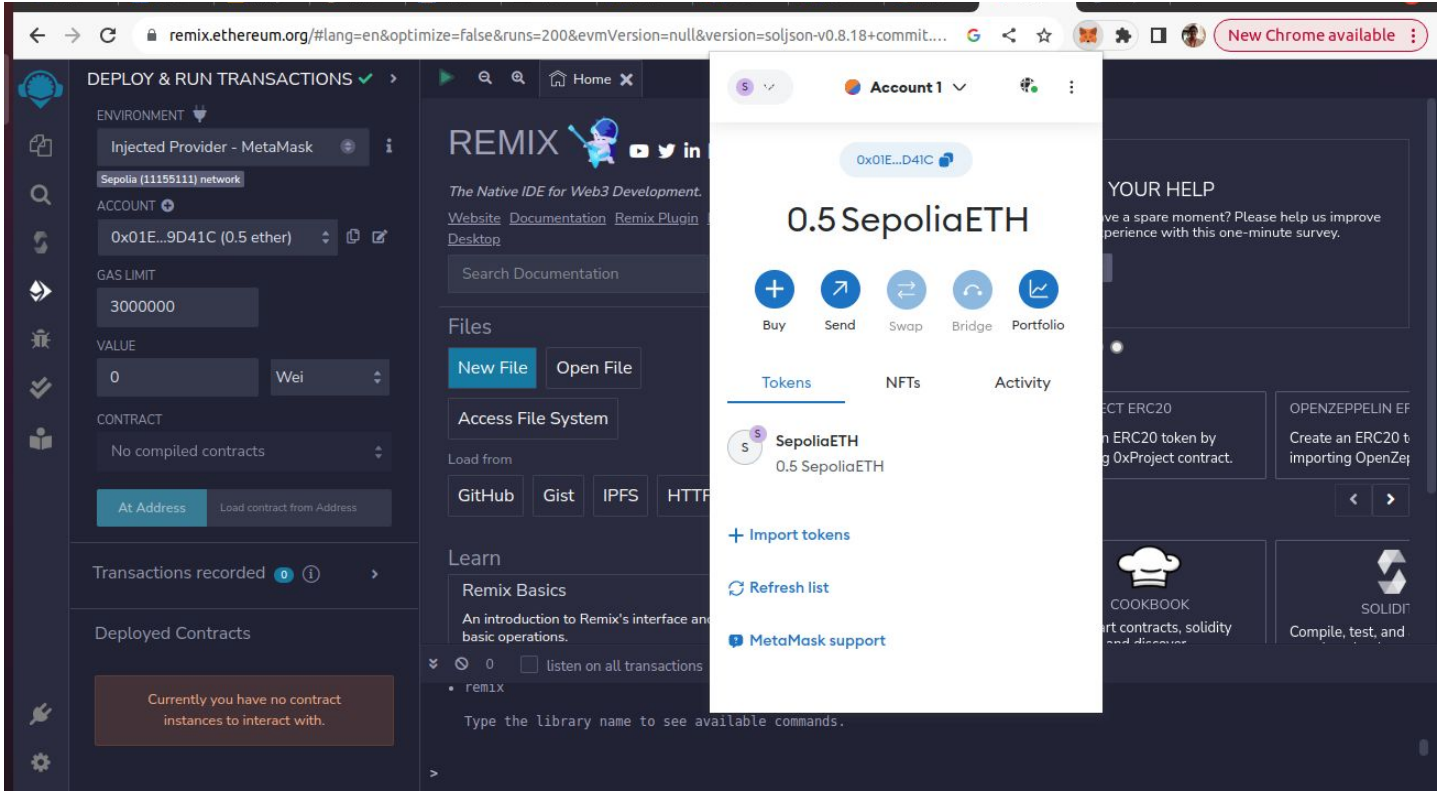
Transferring Ethers using Metamask

Select the Account on the Metamask with which the Remix IDE needs to be connected



Transferring Ethers using Metamask

On the Remix IDE the Account details are displayed

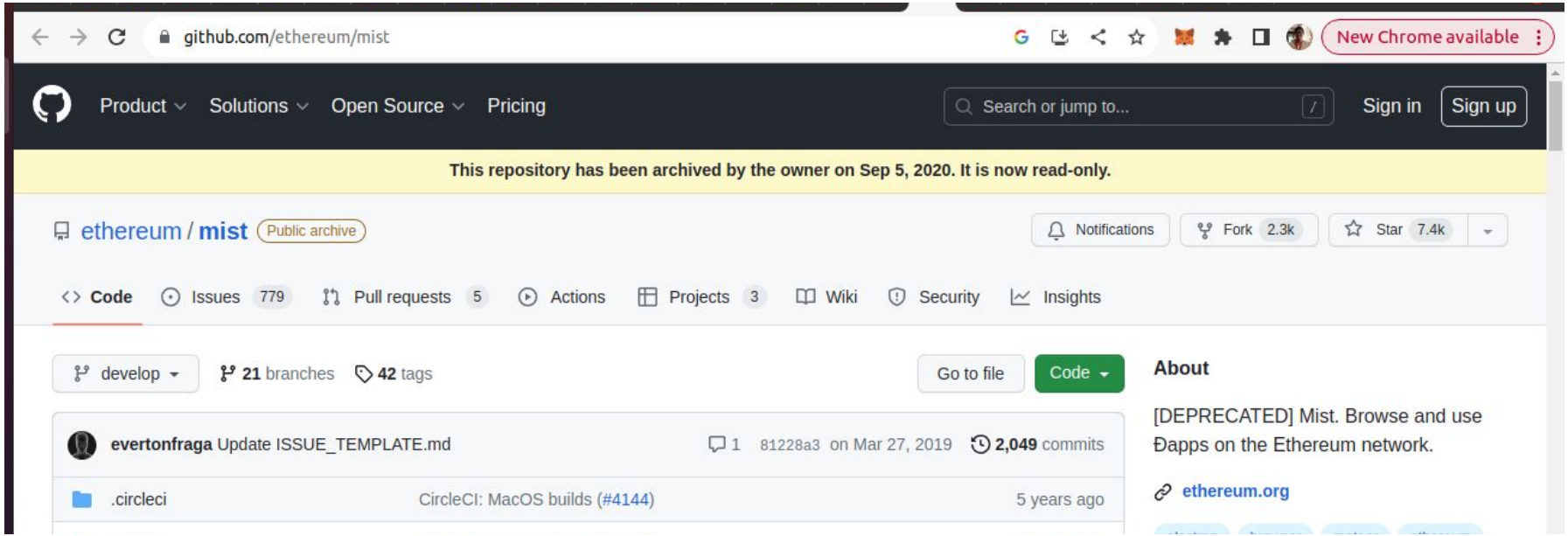


Mist Wallet -- deprecated

- The Ethereum mist was launched in 2017.
- The mist is a blockchain-based browser that serves as a platform for operating blockchain networks and decentralized applications.
- So, the mist is a search engine with several dApps connected to it, and Ethereum is one of these.
- Mist is the official Ethereum wallet and electron application created by the same developers of the Ethereum ecosystem.
- This digital wallet comes in two wallet solutions: the simple wallet and the multi-signature wallet which provides extra security.
- Mist is a hybrid desktop application with a web interface which allows developers to quickly make changes. MyEtherWallet is a Mist application that allows users to access one dApp, which is the official wallet dApp of Ethereum.
- Mist provides users a way to view the Ethereum blockchain and interact with specific components on the blockchain such as Ether, DAO, and smart contracts

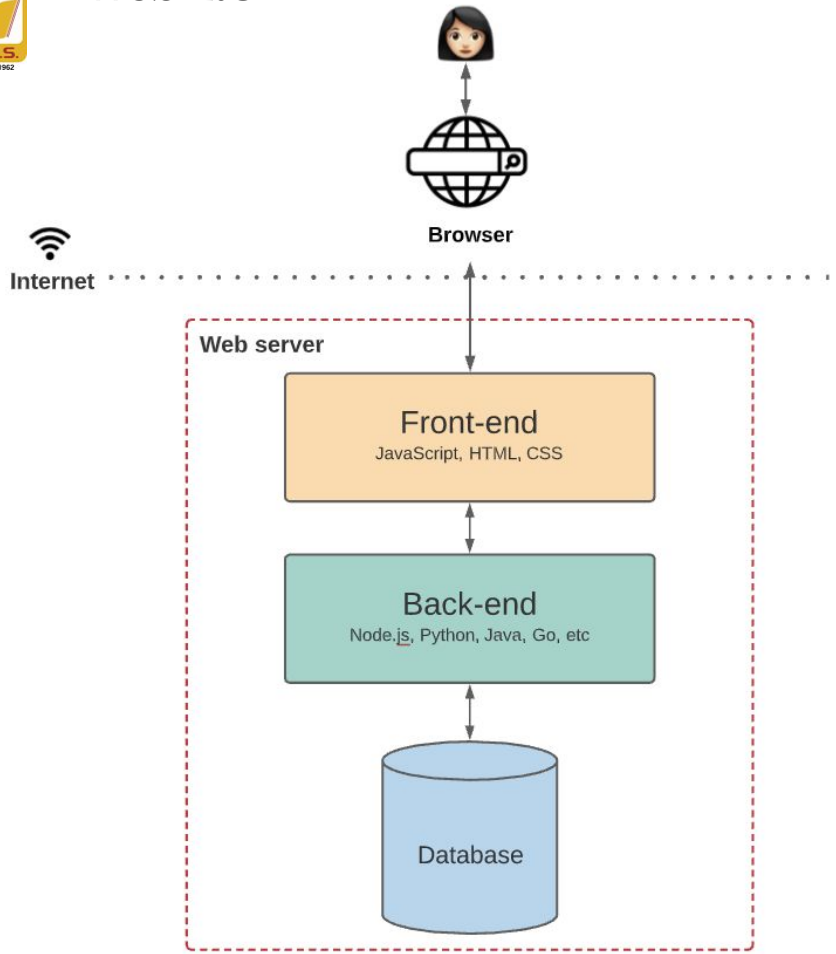
Courtesy : Ethereum - Mist

Mist Wallet -- deprecated

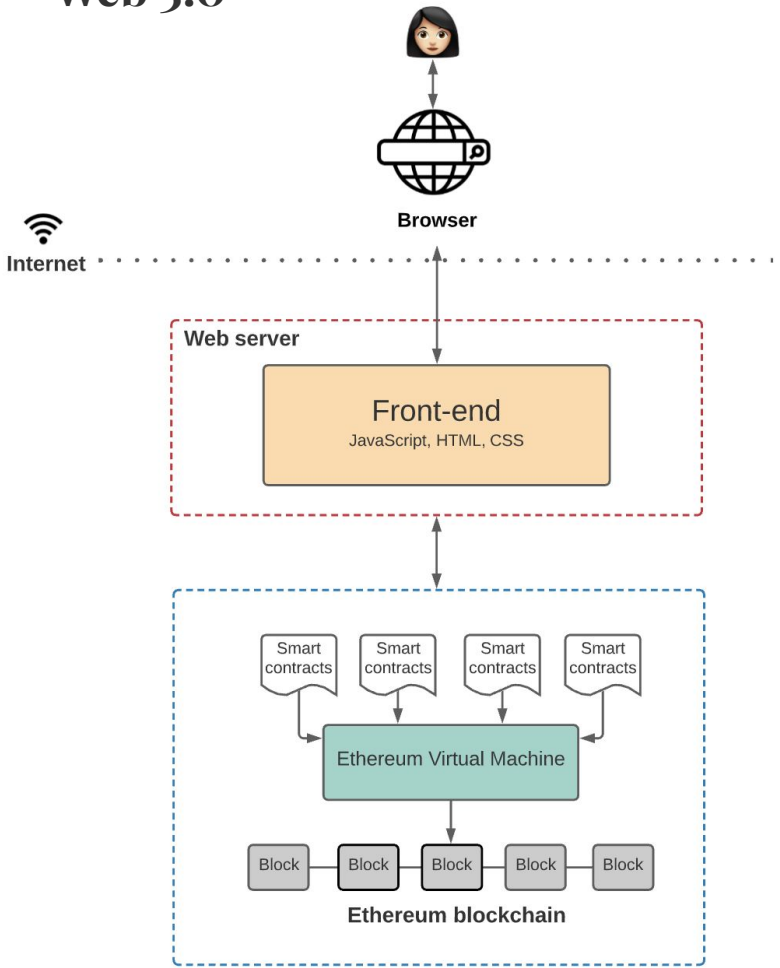


Courtesy : Ethereum - Mist
InsideBitcoins - Mist Wallet

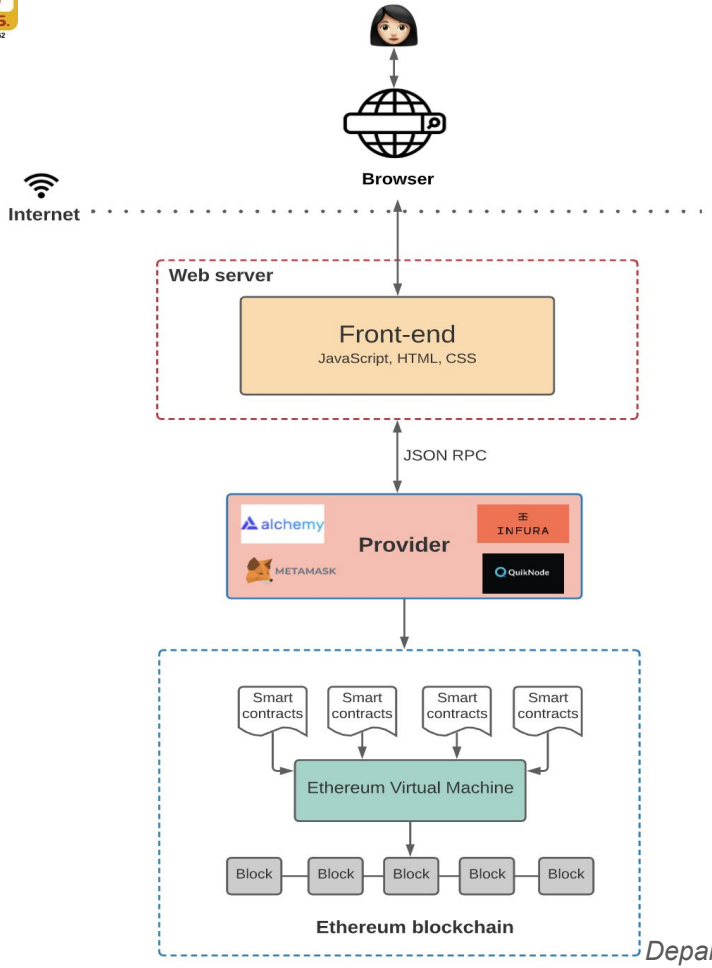
Web 2.0


Browser
Internet
Web server
Front-end
JavaScript, HTML, CSS
Back-end
Node.js, Python, Java, Go, etc
Database

Web 3.0


Browser
Internet
Web server
Front-end
JavaScript, HTML, CSS
Smart contracts
Smart contracts
Smart contracts
Smart contracts
Ethereum Virtual Machine
Block
Block
Block
Block
Block
Ethereum blockchain

Frontend Code Communicate



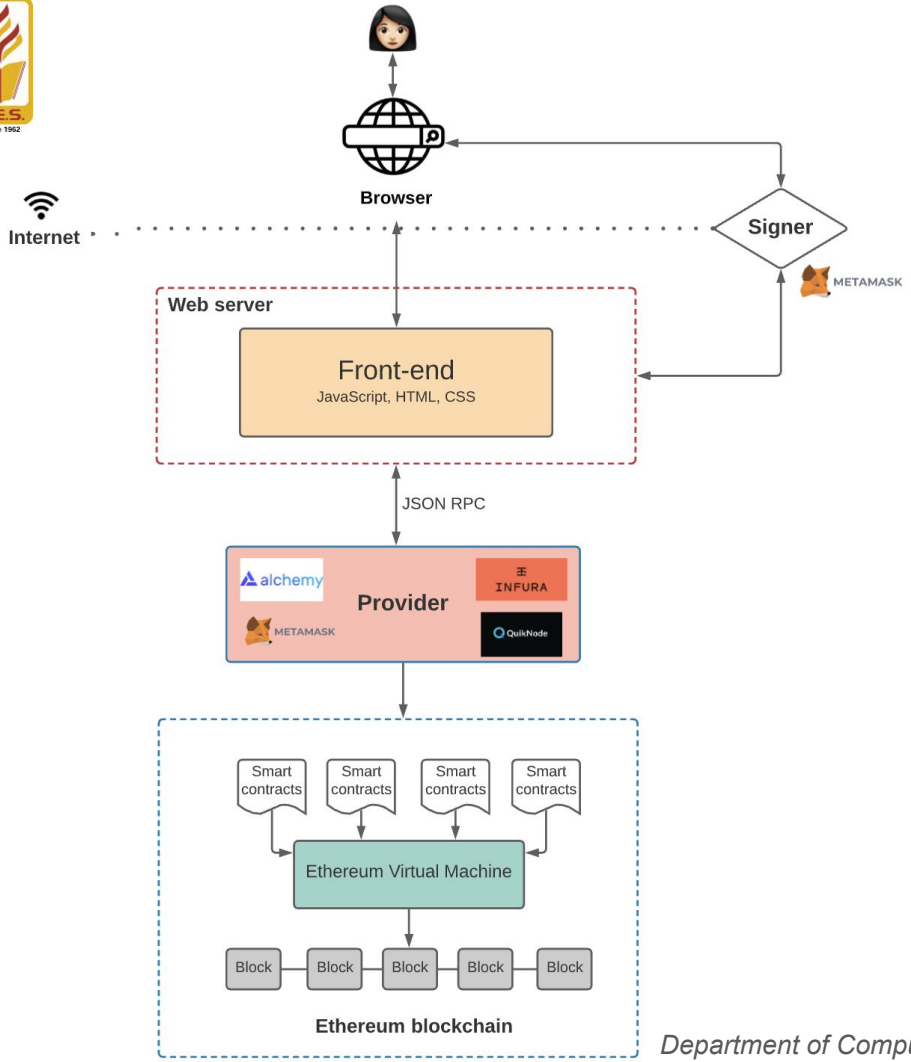
When we want to interact with the data and code on a blockchain, we need to interact with one of these nodes. This is because any node can broadcast a request for a transaction to be executed on the EVM. A miner will then execute the transaction and propagate the resulting state change to the rest of the network.

There are two ways to broadcast a new transaction:

1. **Set up your own node which runs the Ethereum blockchain software**
2. **Use nodes provided by third-party services like Infura, Alchemy, and Quicknode**

Every Ethereum client (i.e. provider) implements a JSON-RPC specification. This ensures that there's a uniform set of methods when frontend applications want to interact with the blockchain.

Frontend Code Communicate



When a user wants to publish a new post onto the chain, our DApp would ask the user to “sign” the transaction using their private key — only then would the DApp relay the transaction to the blockchain. Otherwise, the nodes wouldn’t accept the transaction.

This “signing” of transactions is where Metamask typically comes in.

Storage on the Blockchain

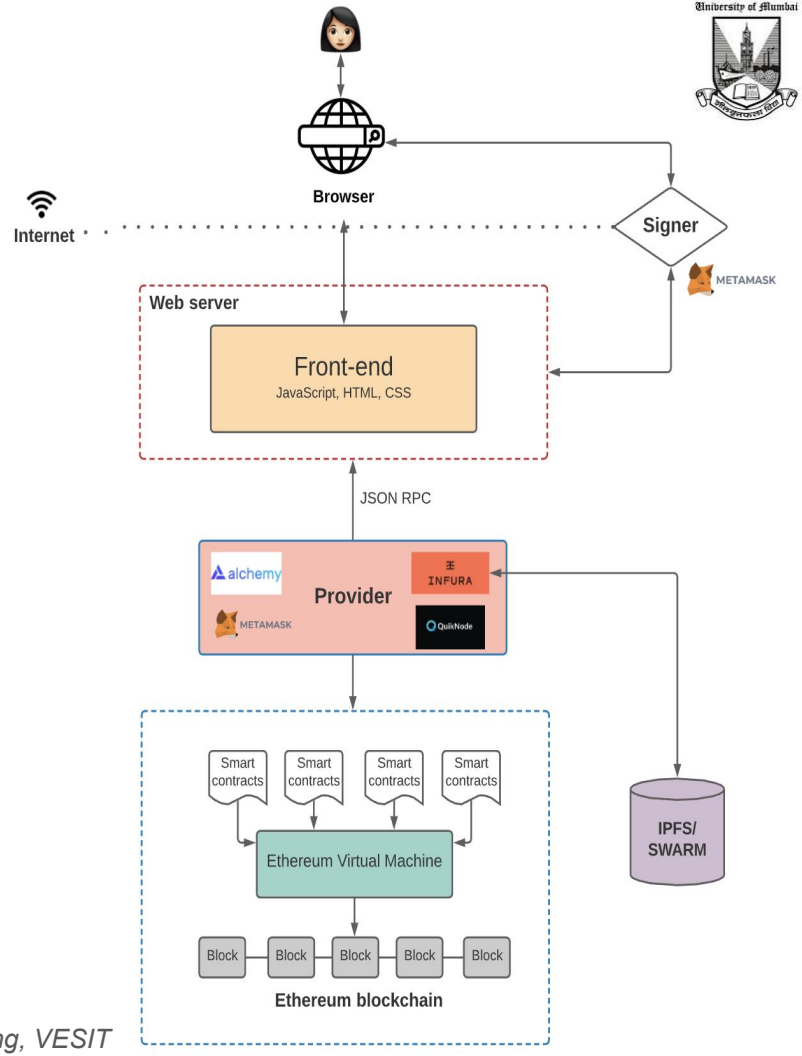
Storing everything on the blockchain gets really expensive,

Users to pay extra for using your DApp every time their transaction requi adding a new state is not the best user experience. One way to combat this to use a decentralized off-chain storage solution, like IPFS or Swarm.

IPFS is a distributed file system for storing and accessing data. So, rather th storing data in a centralized database, the IPFS system distributes and sto the data in a peer-to-peer network. This makes it easy for you to retrieve when you need to.

IPFS also has an incentive layer known as “Filecoin.” This layer incentiviz nodes around the world to store and retrieve this data. You can use a provic like Infura (which provides you with an IPFS node) or Pinata (which provides easy-to-use service where you can “pin” your files to IPFS and take the IP hash and store that on the blockchain).

Swarm is similar in that it’s a decentralized storage network, but there’s c notable difference. While Filecoin is a separate system, Swarm’s incent system is built-in and enforced through smart contracts on the Ethere blockchain for storing and retrieving data.

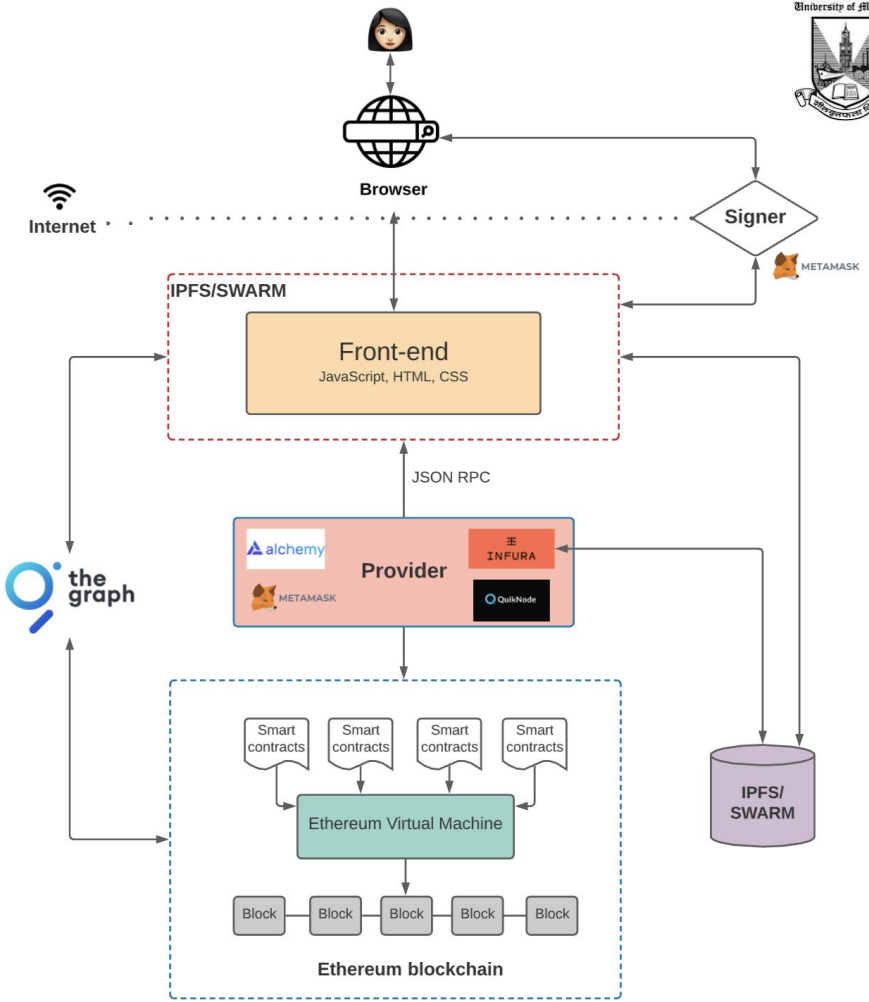


Querying the Blockchain

Smart Contract Events

You can use the Web3.js library to query and listen for smart contract events. You can listen to specific events and specify a callback every time the event is fired. For instance, if you have a smart contract that sends a continuous payment stream from person A to person B every block, then you can emit an event every time a new payment is made to person B. Your frontend code can listen to events being fired by the smart contract and carry out specific actions based on it.

The Graph is an off-chain indexing solution that makes it easier to query data on the Ethereum blockchain. The Graph allows you to define which smart contracts to index, which events and function calls to listen to, and how to transform incoming events into entities that your frontend logic (or whatever is using the API) can consume. It uses GraphQL as a query language, which many frontend engineers love because of how expressive it is compared to traditional REST APIs.



Ethereum frameworks

- Building a full-fledged dapp requires different pieces of technology. Software frameworks include many of the needed features or provide easy plugin systems to pick the tools you desire.
- Frameworks come with a lot of out-of-the-box functionality, like:
  - Features to spin up a local blockchain instance.
  - Utilities to compile and test your smart contracts.
  - Client development add-ons to build your user-facing application within the same project/repository.
  - Configuration to connect to Ethereum networks and deploy contracts, whether to a locally running instance, or one of Ethereum's public networks.
  - Decentralized app distribution - integrations with storage options like IPFS.

Ethereum frameworks

- Truffle - A development environment, testing framework, build pipeline, and other tools.
- Hardhat - Ethereum development environment for professionals.
- Ape - The smart contract development tool for Pythonistas, Data Scientists, and Security Professionals.
- Brownie - Python-based development environment and testing framework.
- Web3j - A platform for developing blockchain applications on the JVM.
- OpenZeppelin SDK - The Ultimate Smart Contract Toolkit: to help you develop, compile, upgrade, deploy and interact with smart contracts.
- Create Eth App - Create Ethereum-powered apps with one command. Comes with a wide offering of UI frameworks and DeFi templates to choose from.
- Scaffold-Eth - Ethers.js + Hardhat + React components and hooks for web3
- Tenderly - Web3 development platform that enables blockchain developers to build, test, debug, monitor, and operate smart contracts and improve dapp UX.
- The Graph - The Graph for querying blockchain data efficiently.
- Alchemy - Ethereum Development Platform.
- Foundry - A blazing fast, portable and modular toolkit for Ethereum application development written in Rust.
- NodeReal - Ethereum Development Platform.
- thirdweb SDK - Build web3 applications that can interact with your smart contracts using our powerful SDKs and CLI.
- Chainstack - Web3 (Ethereum and otherwise) Development Platform.

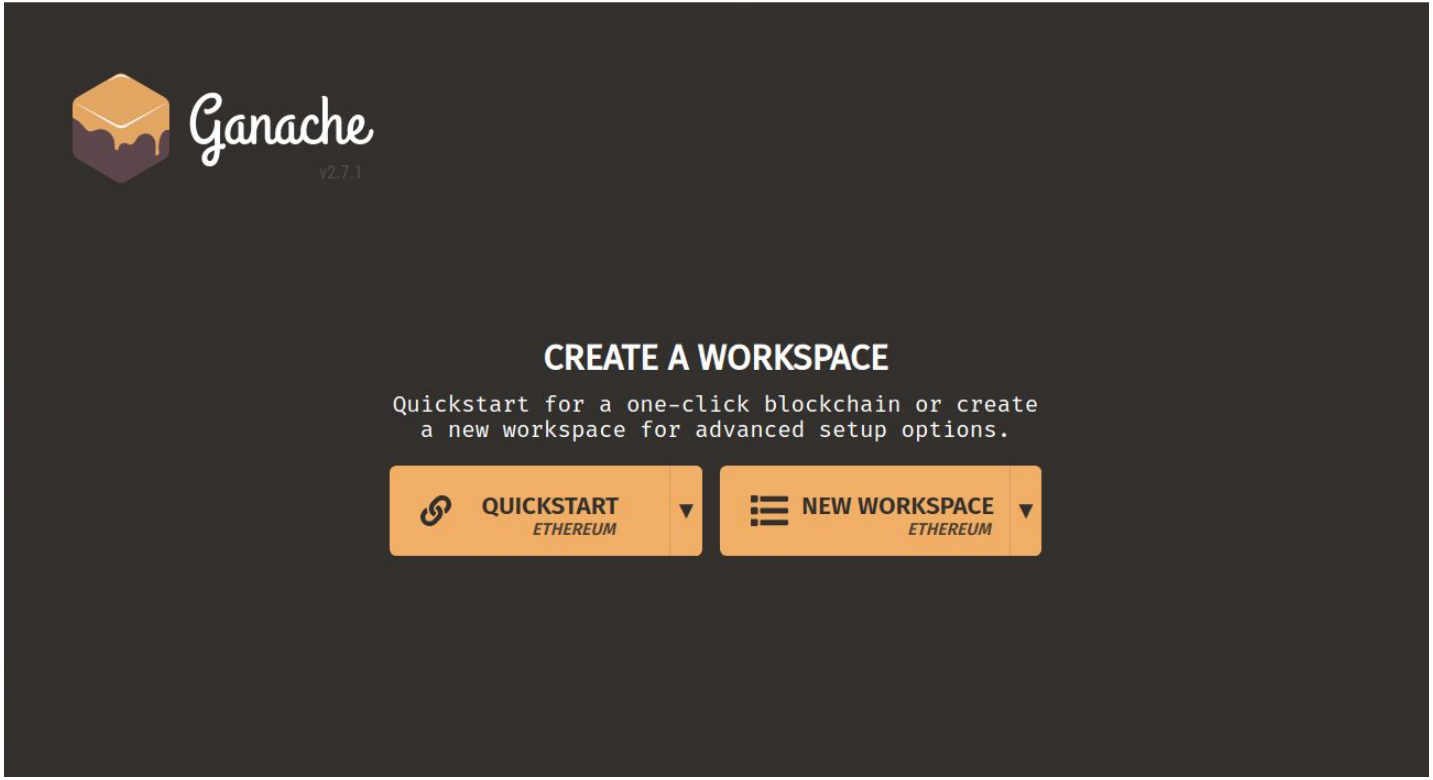
Case study of Ganache for Ethereum blockchain

- Ganache is a personal blockchain for rapid Ethereum and Filecoin distributed application development.
- One can use Ganache across the entire development cycle;
- enabling you to develop, deploy, and test your dApps in a safe and deterministic environment.
- Ganache comes in two flavors: a UI and CLI.
    - Ganache UI is a desktop application supporting Ethereum and Filecoin technology.
    - More robust command-line tool, ganache, is available for Ethereum development. It offers:
        - console.log in Solidity
        - Zero-config Mainnet and testnet forking
        - Fork any Ethereum network without waiting to sync
        - Ethereum JSON-RPC support
        - Snapshot/revert state
        - Mine blocks instantly, on demand, or at an interval
        - Fast-forward time
        - Impersonate any account (no private keys required!)
        - Listens for JSON-RPC 2.0 requests over HTTP/WebSockets
        - Programmatic use in Node.js
        - Pending Transactions

All versions of Ganache are available for Windows, Mac, and Linux.

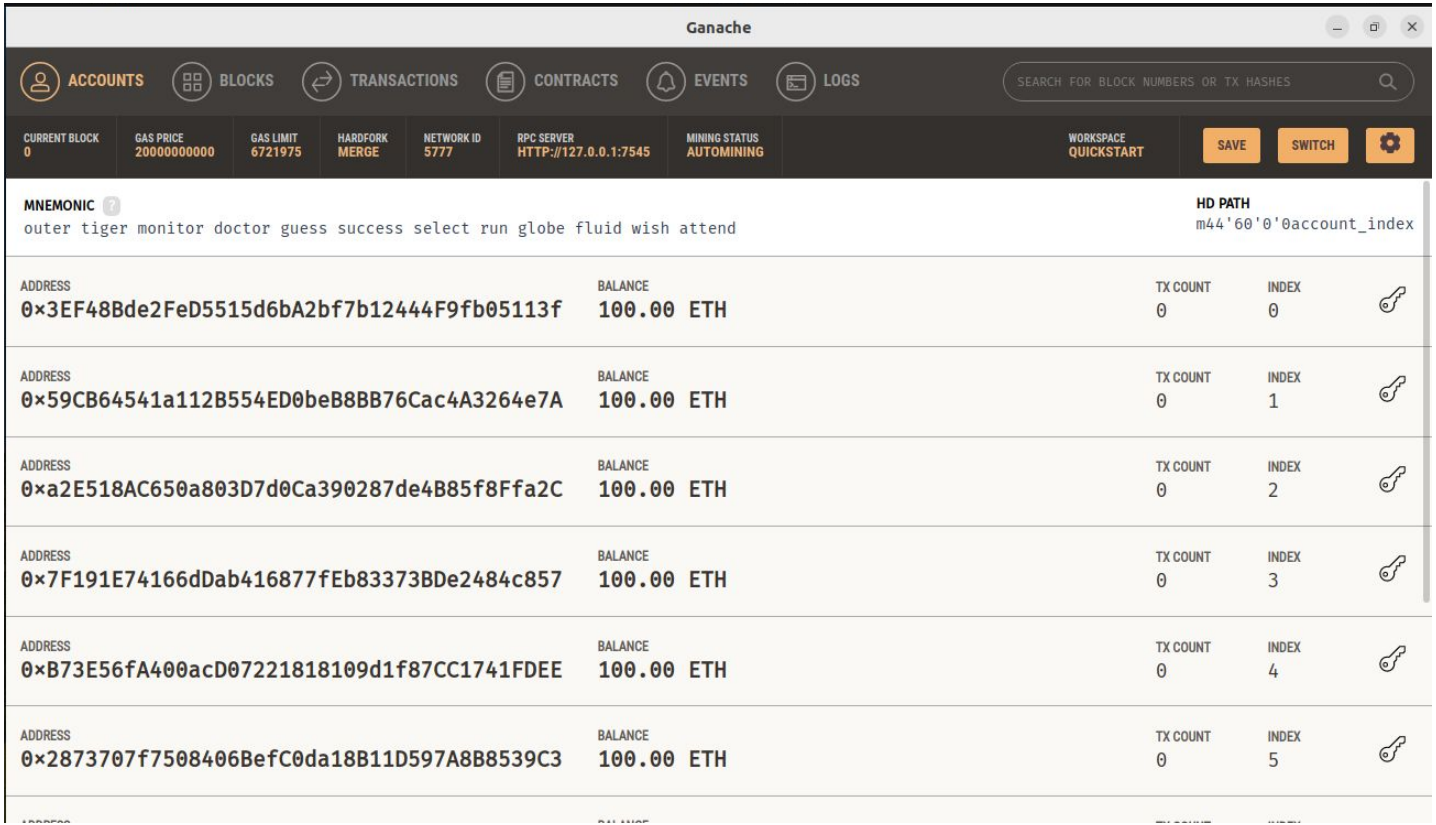
Case study of Ganache for Ethereum blockchain

Flash screen of the Ganache after installation, Click on **Quickstart** to create a blockchain



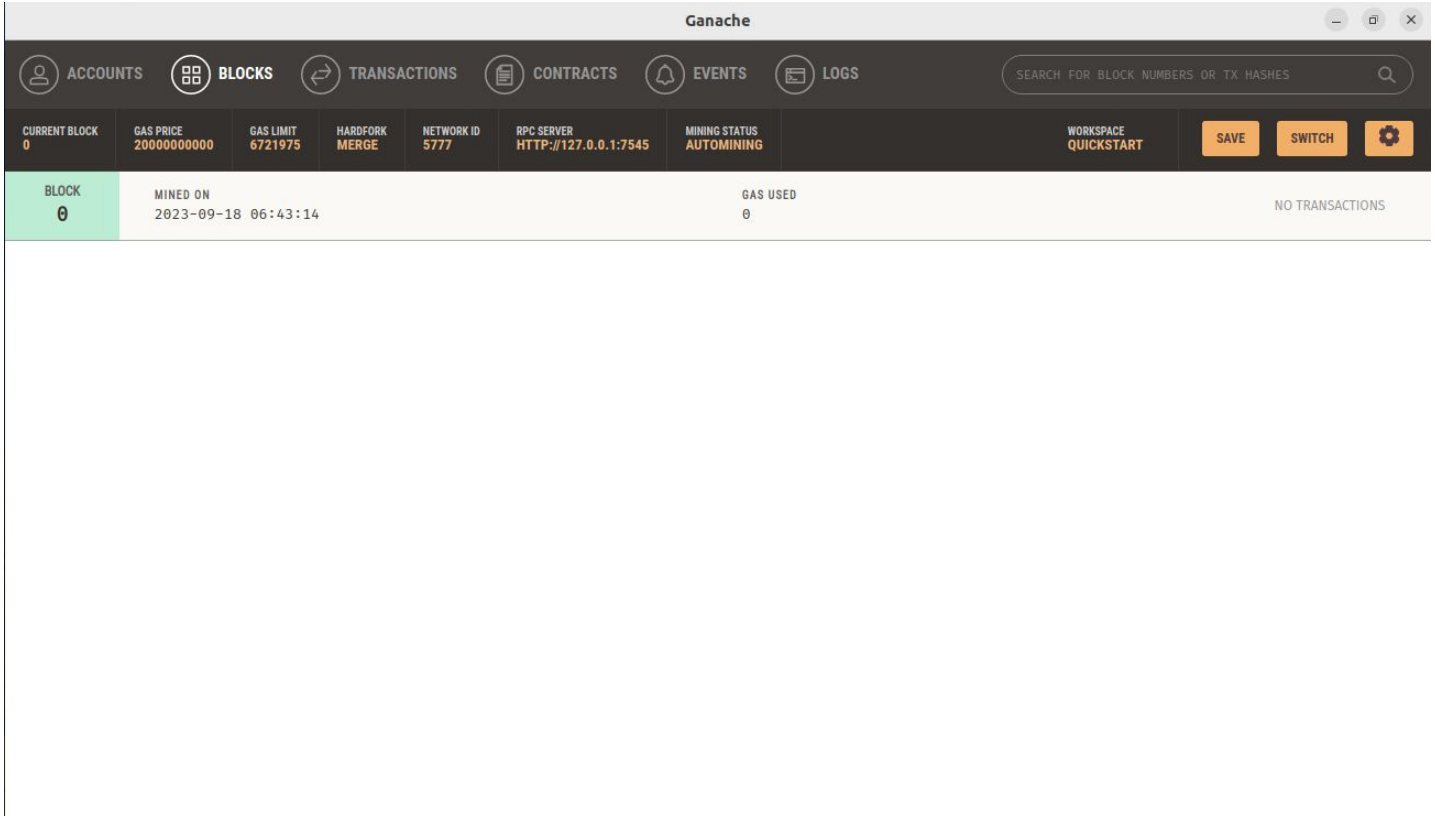
Case study of Ganache for Ethereum blockchain

On the Accounts tab, enlists the 10 Accounts created each with 100 Ethers



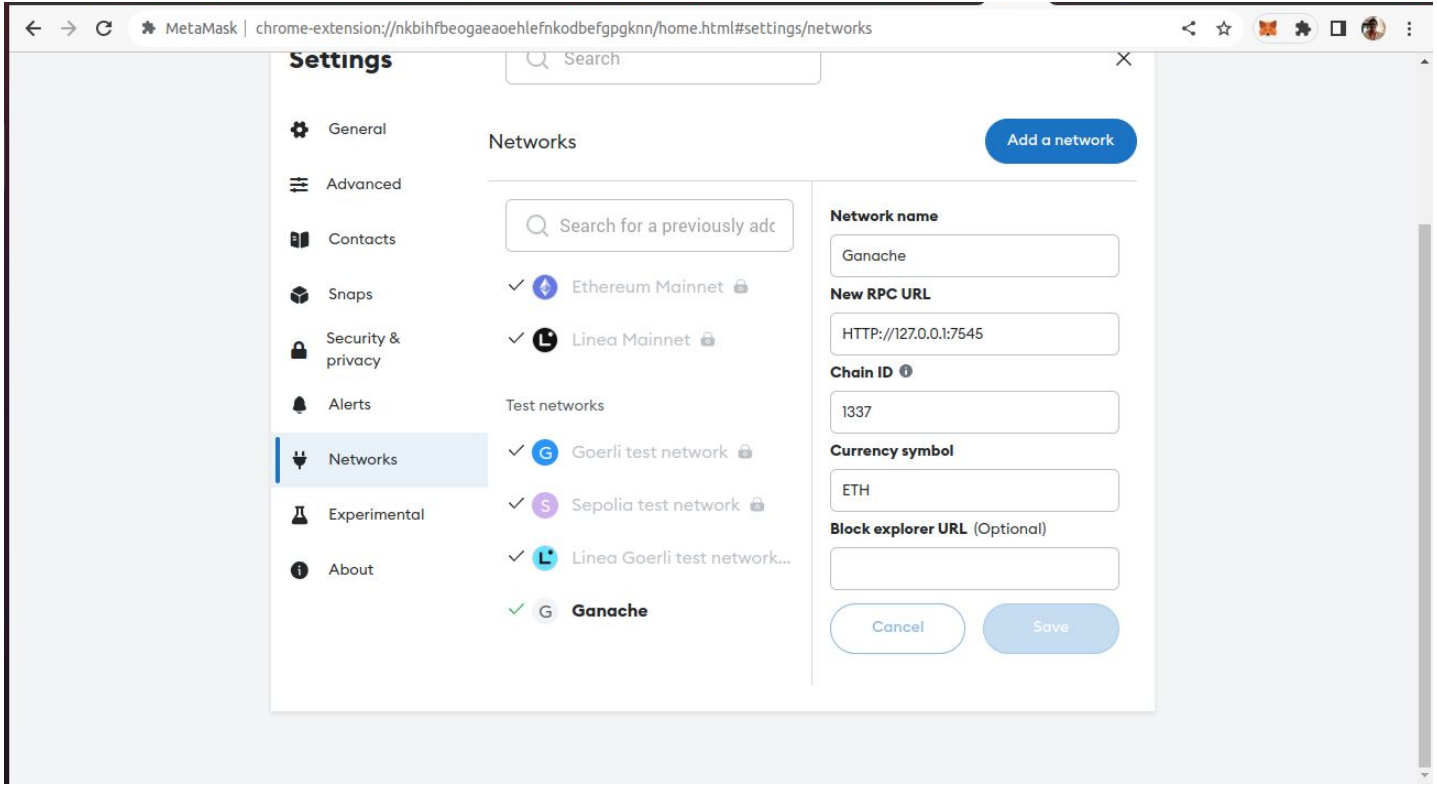
Case study of Ganache for Ethereum blockchain

On the Blocks Tab, we could see the Genesis Block created



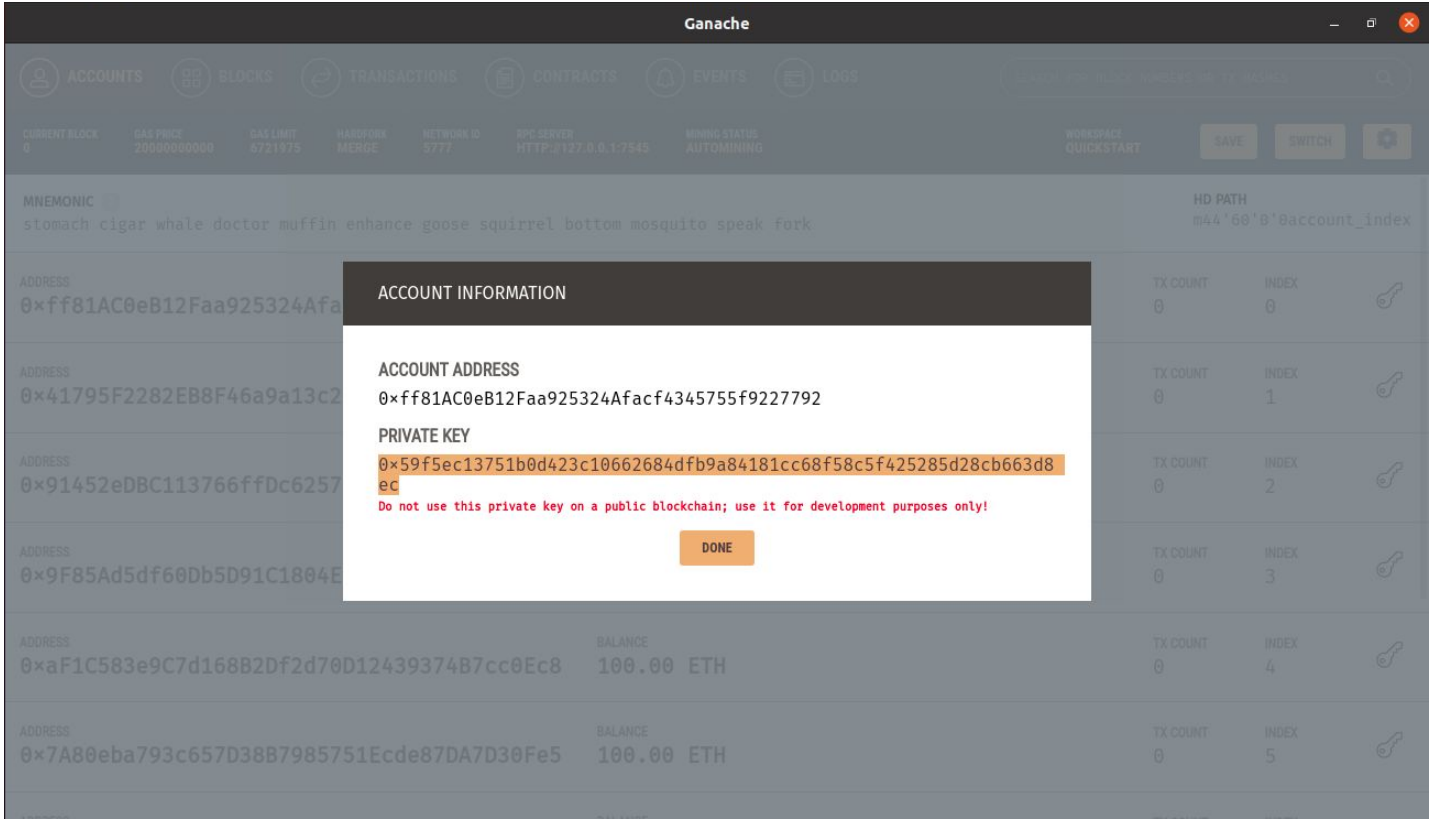
Case study of Ganache for Ethereum blockchain

On the Metamask, add Ganache Network as follows:



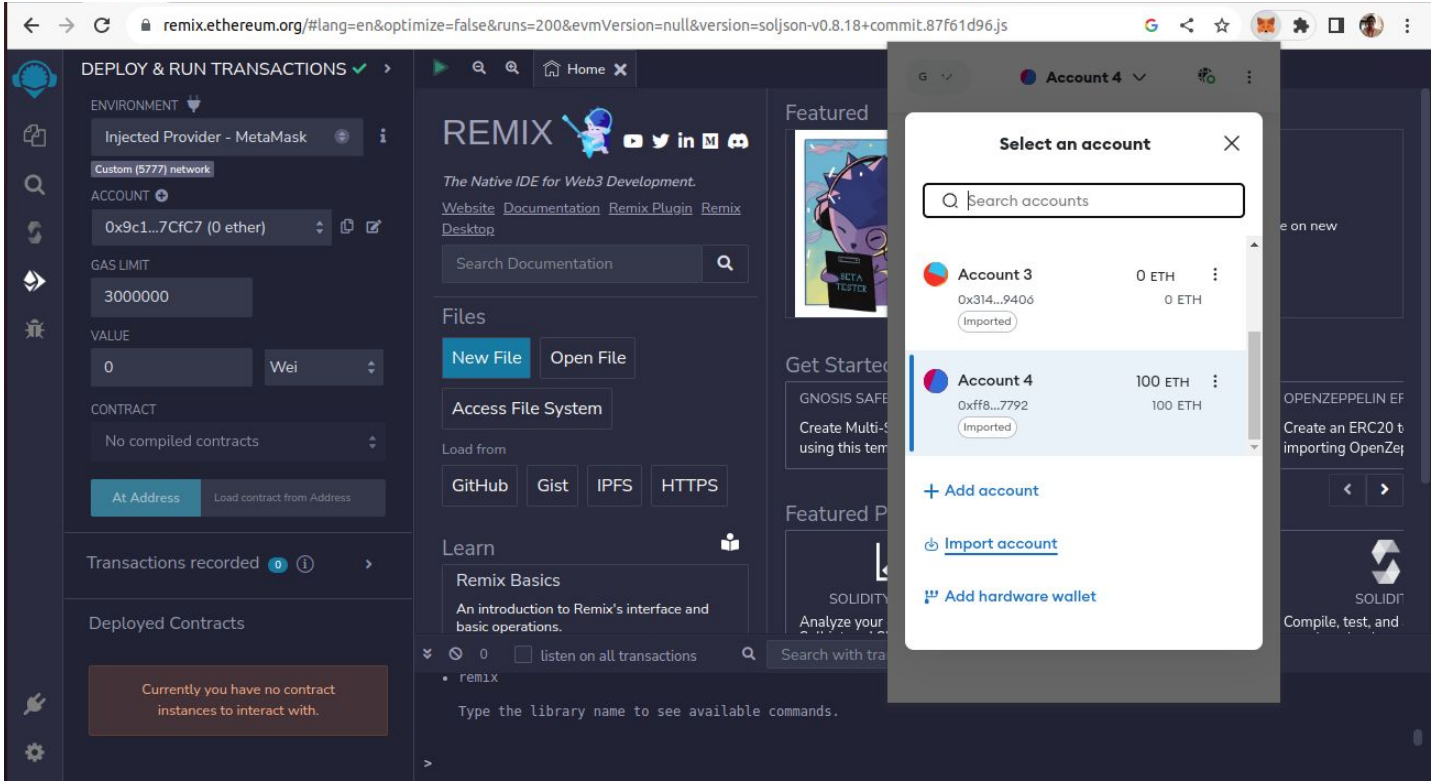
Case study of Ganache for Ethereum blockchain

From on account, copy the Private key



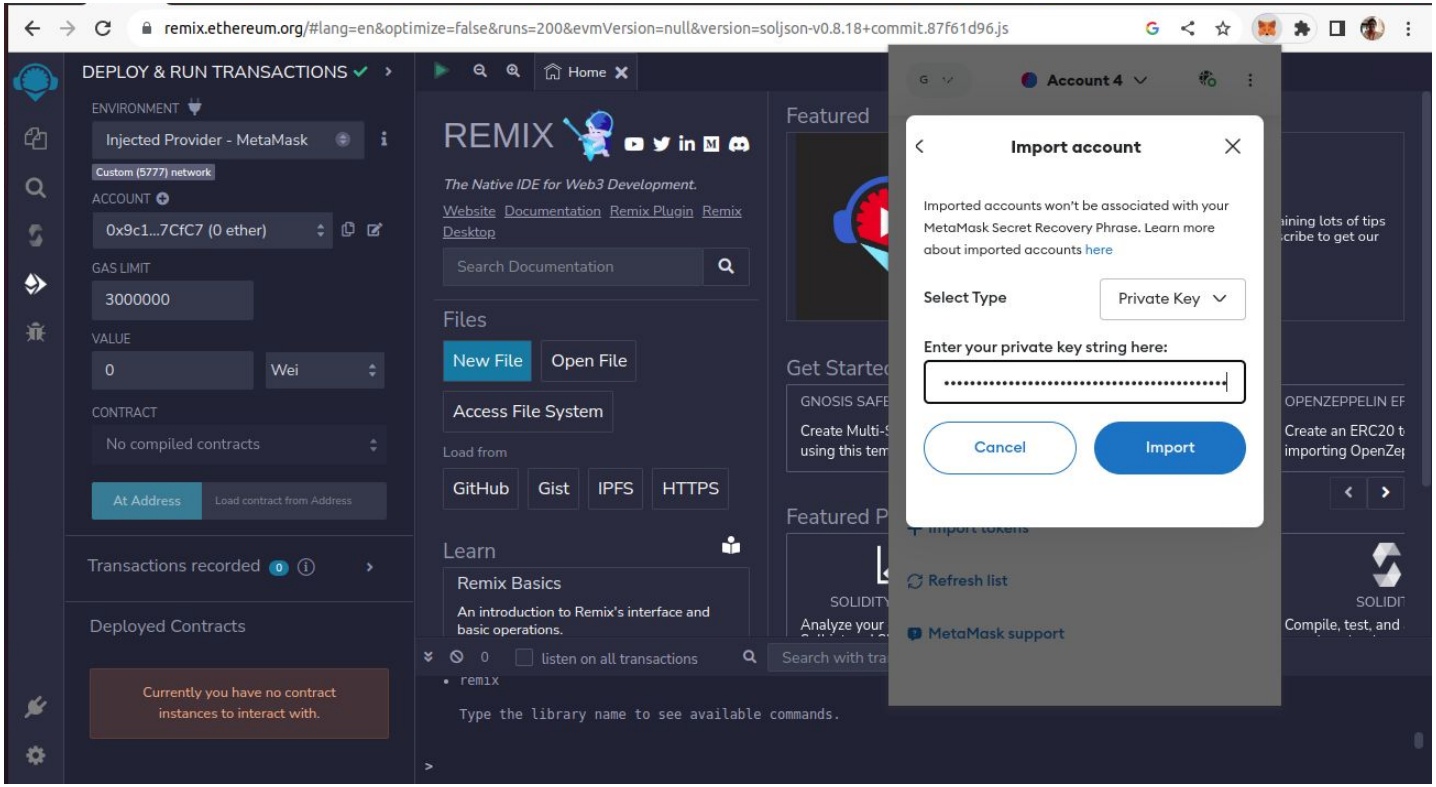
Case study of Ganache for Ethereum blockchain

On the Metamask, select **Import Account**



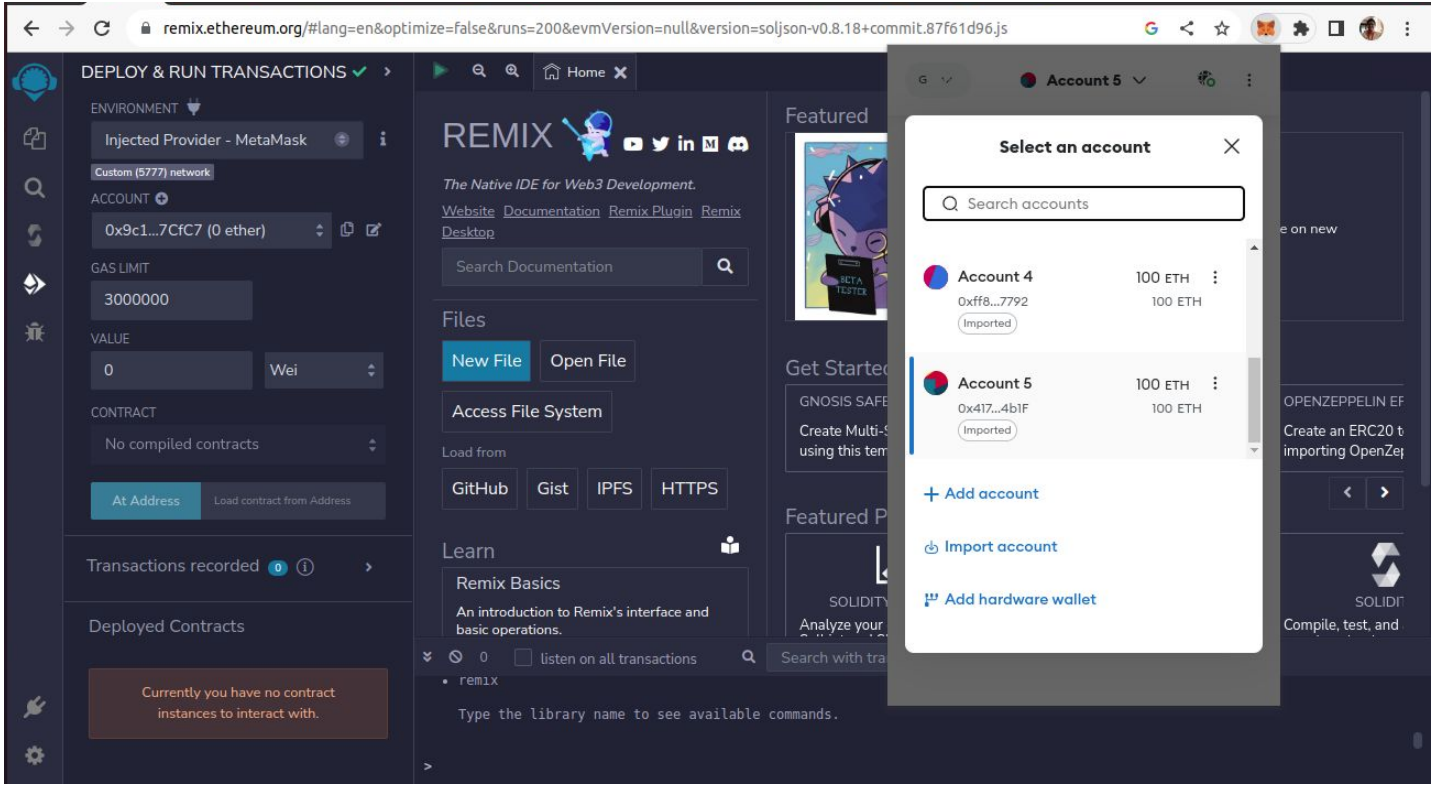
Case study of Ganache for Ethereum blockchain

While importing Account on the Metamask, paste the Private key



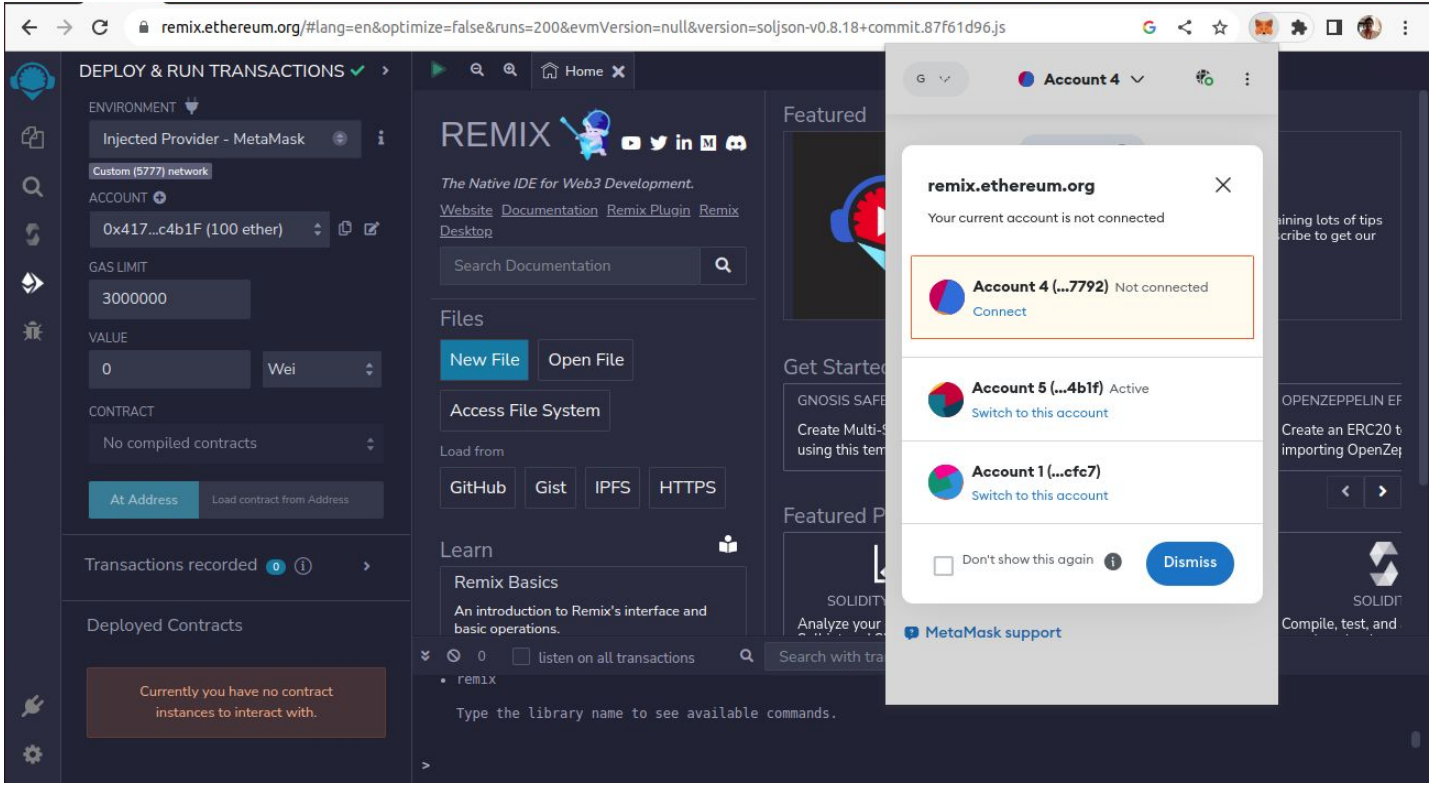
Case study of Ganache for Ethereum blockchain

Enlists the Account imported



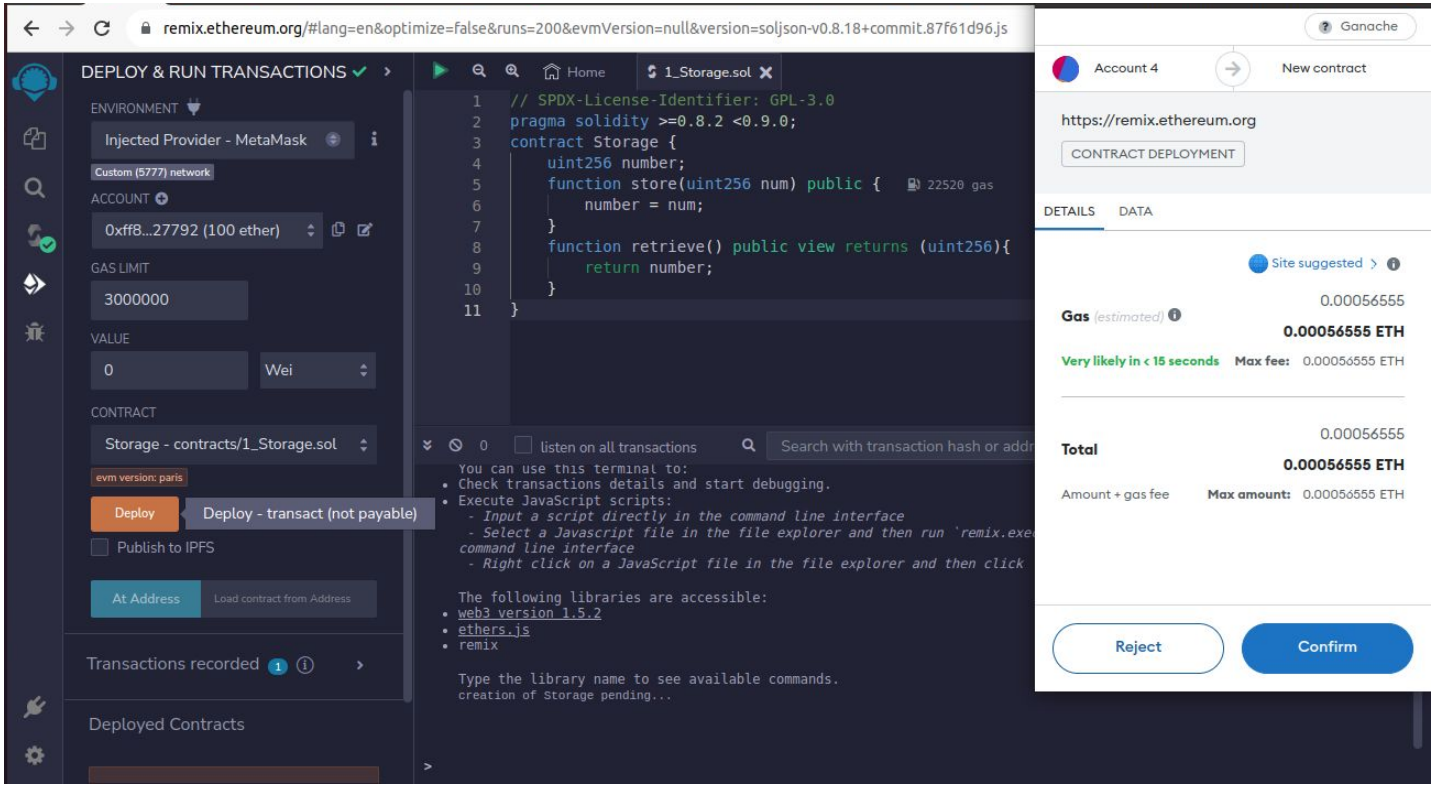
Case study of Ganache for Ethereum blockchain

Select the Account from Metamask to be connected with Remix IDE



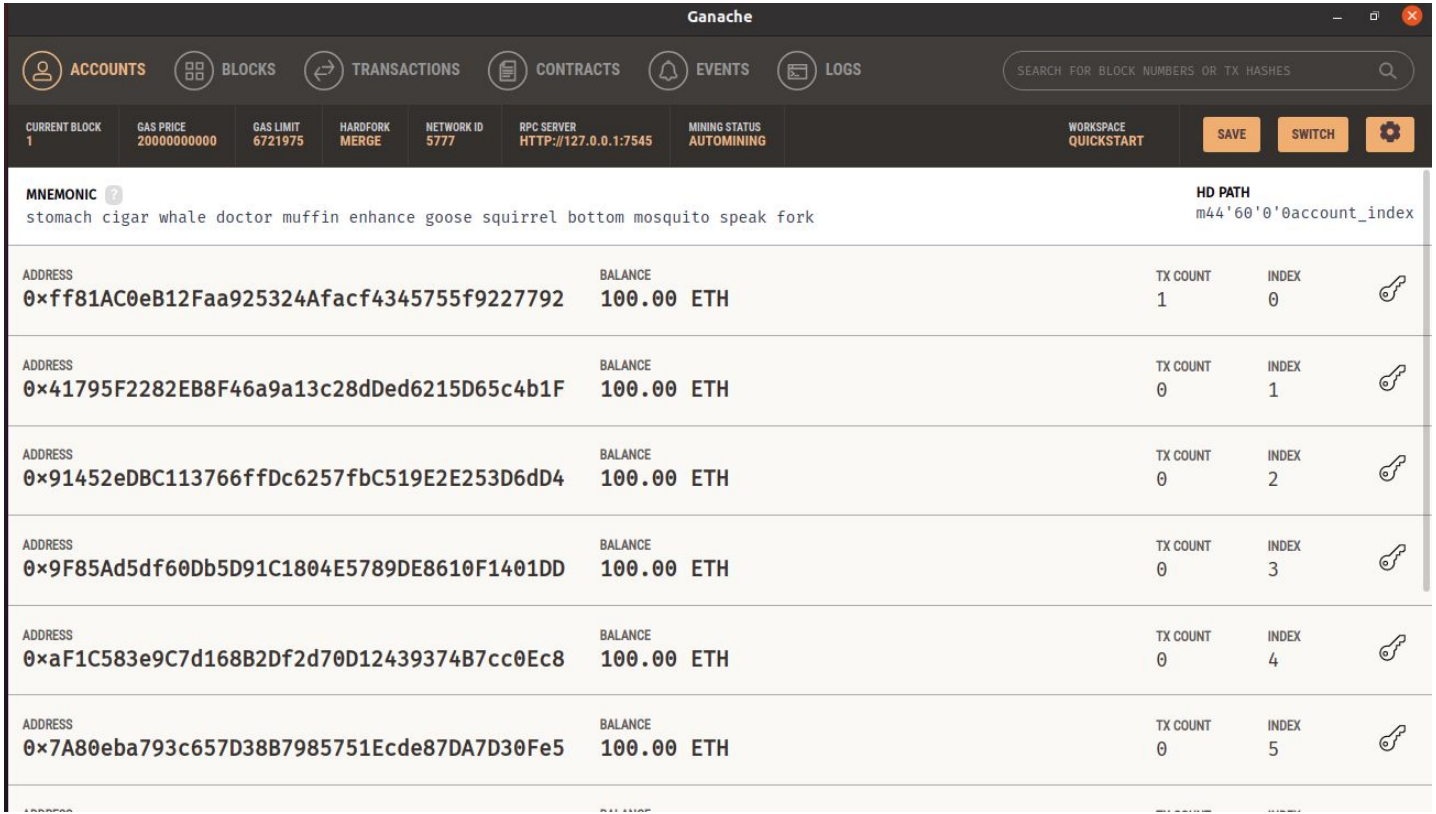
Case study of Ganache for Ethereum blockchain

Deploy the Smart Contract



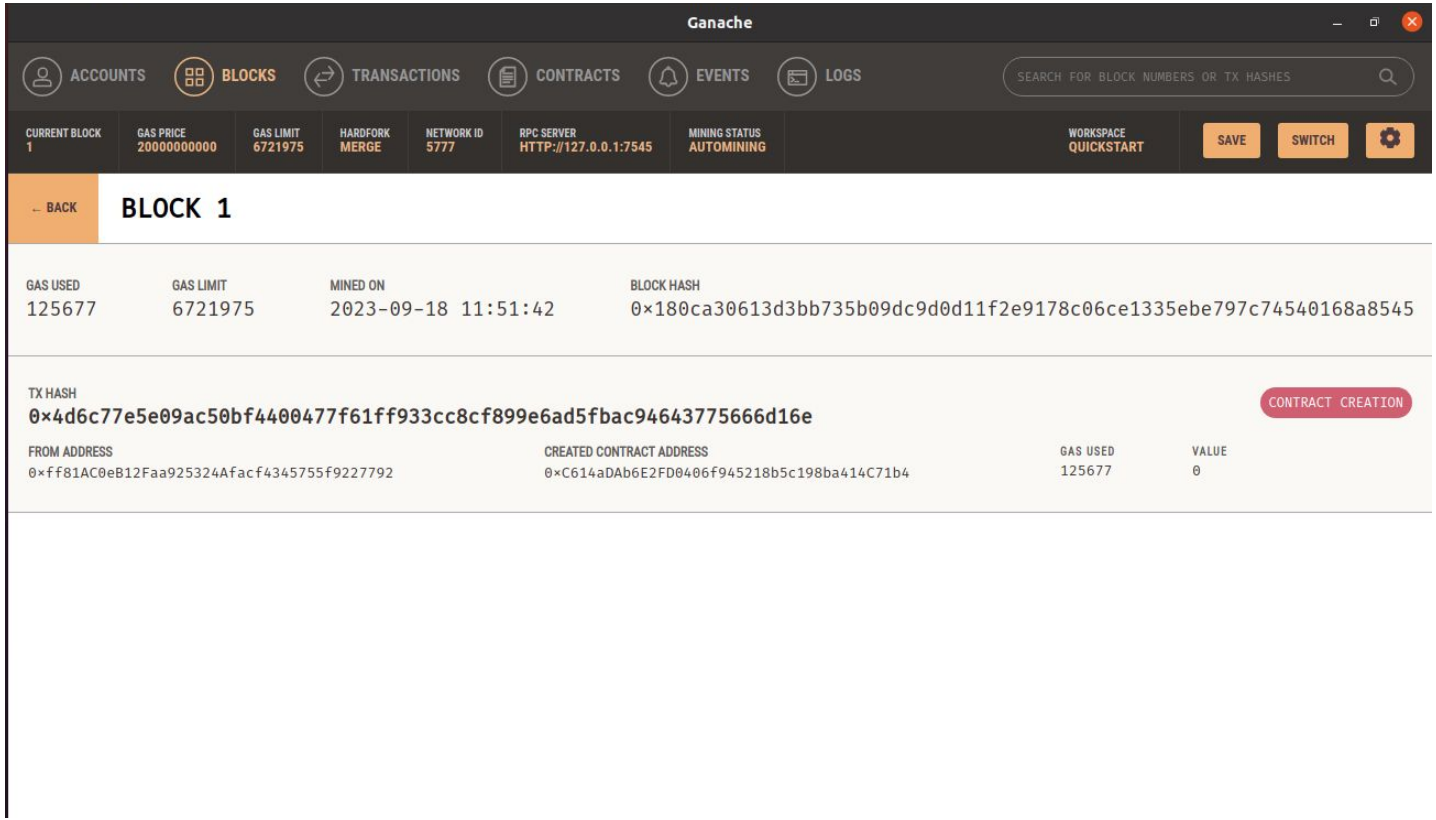
Case study of Ganache for Ethereum blockchain

On the Ganache Environment, the Transaction count is updated



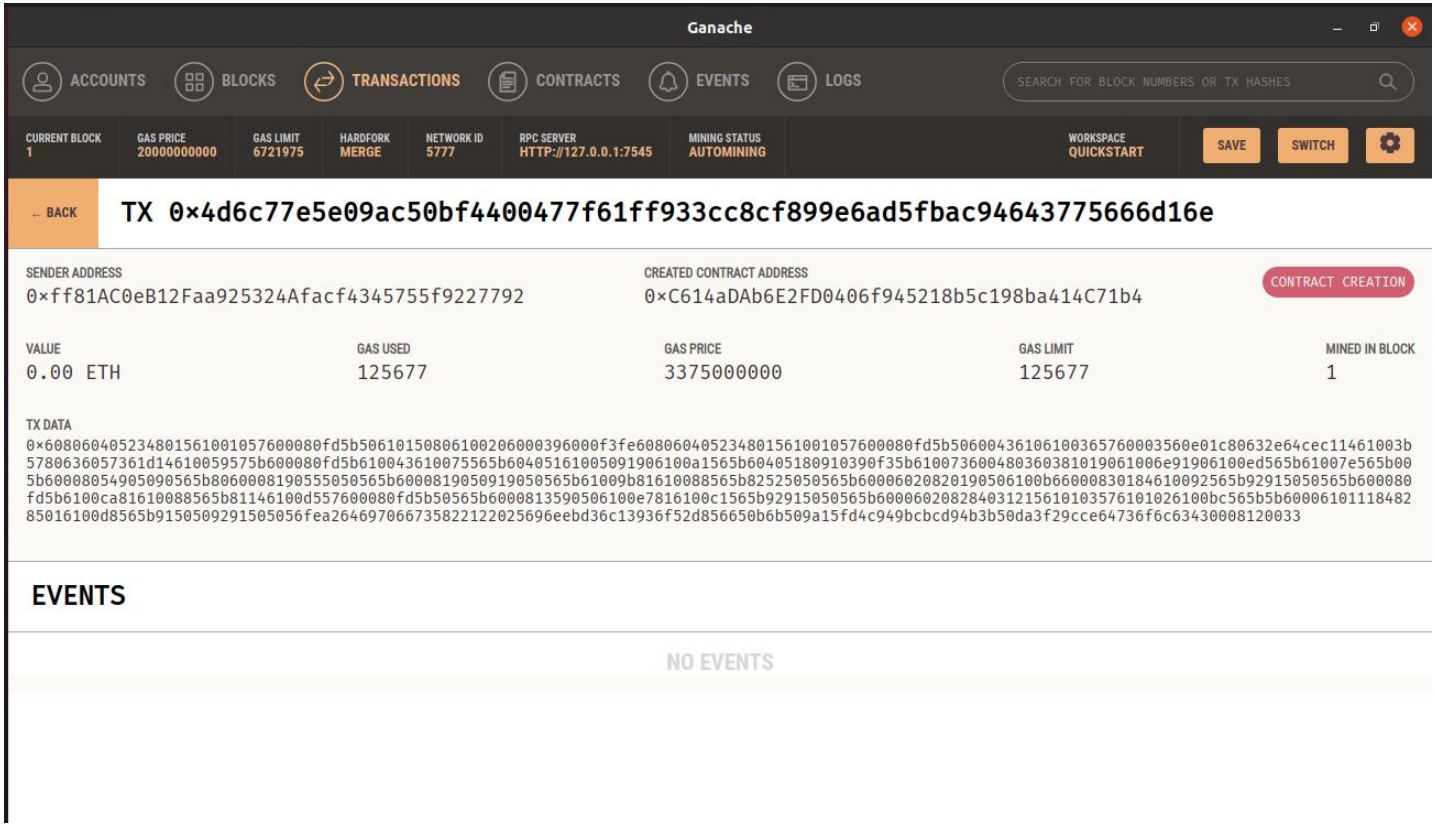
Case study of Ganache for Ethereum blockchain

Block 1 is added to the Blockchain which displays the Contract Created



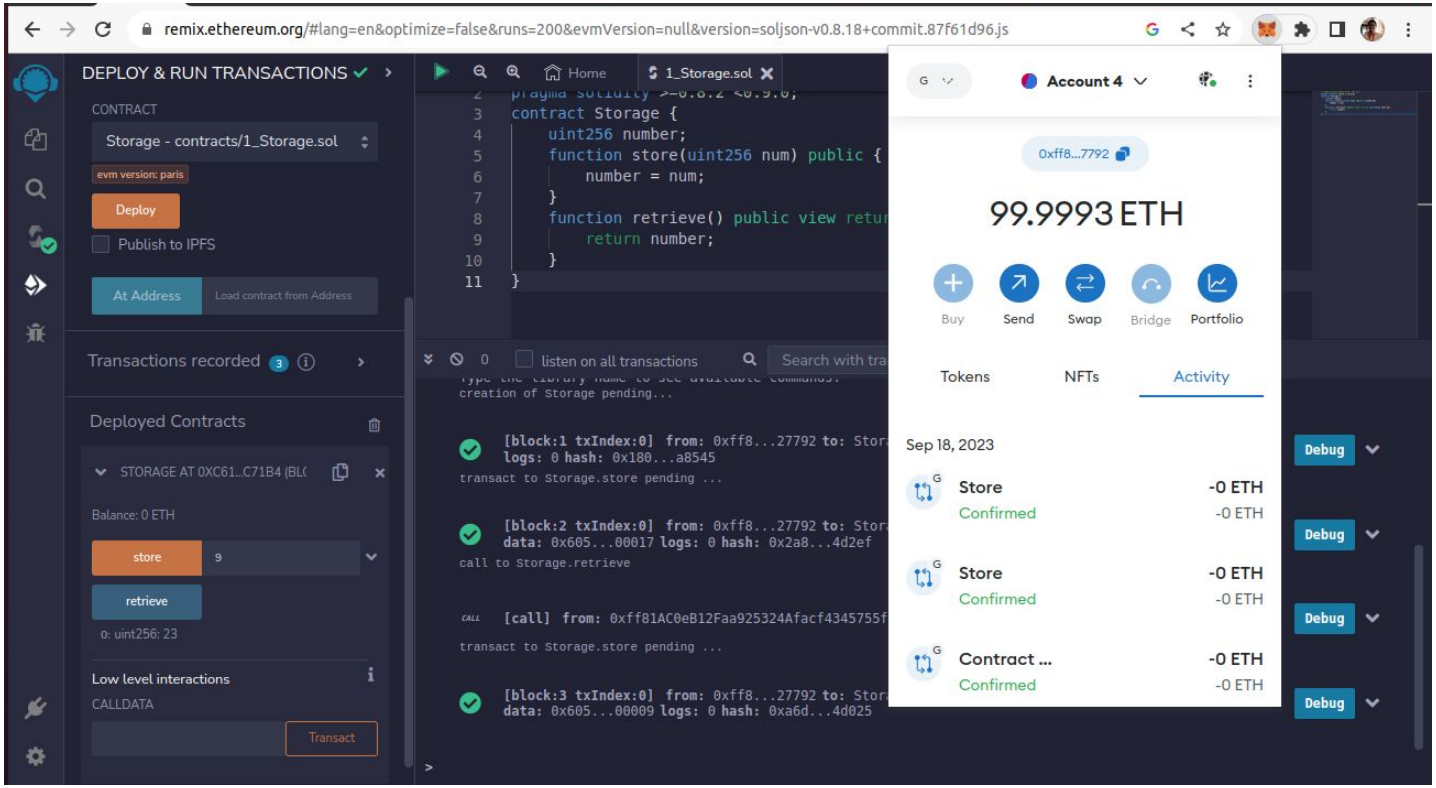
Case study of Ganache for Ethereum blockchain

Transaction details of the Contract is displayed on Ganache Environment



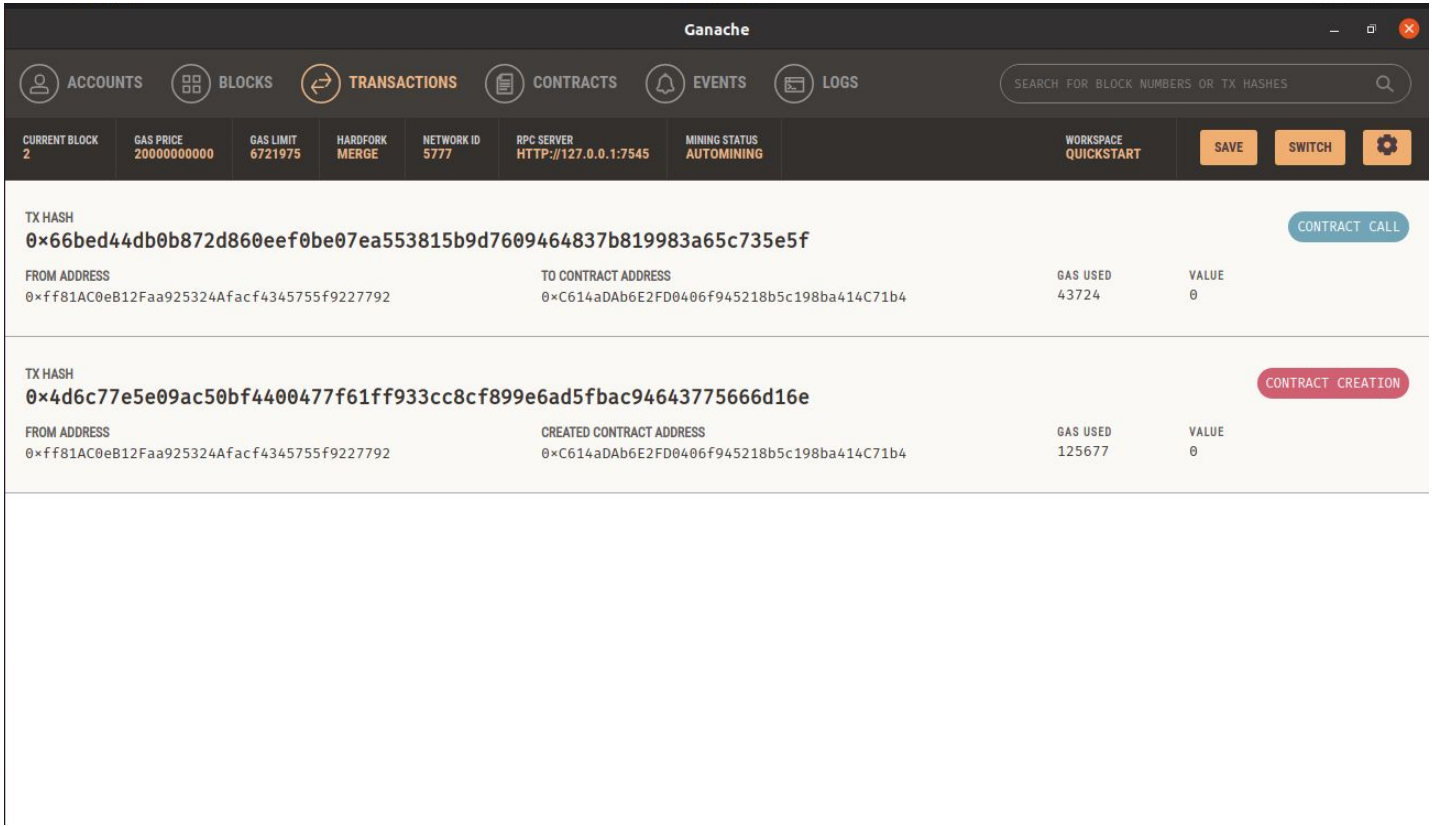
Case study of Ganache for Ethereum blockchain

After interacting with the Smart Contract, funds are updated on the Metamask

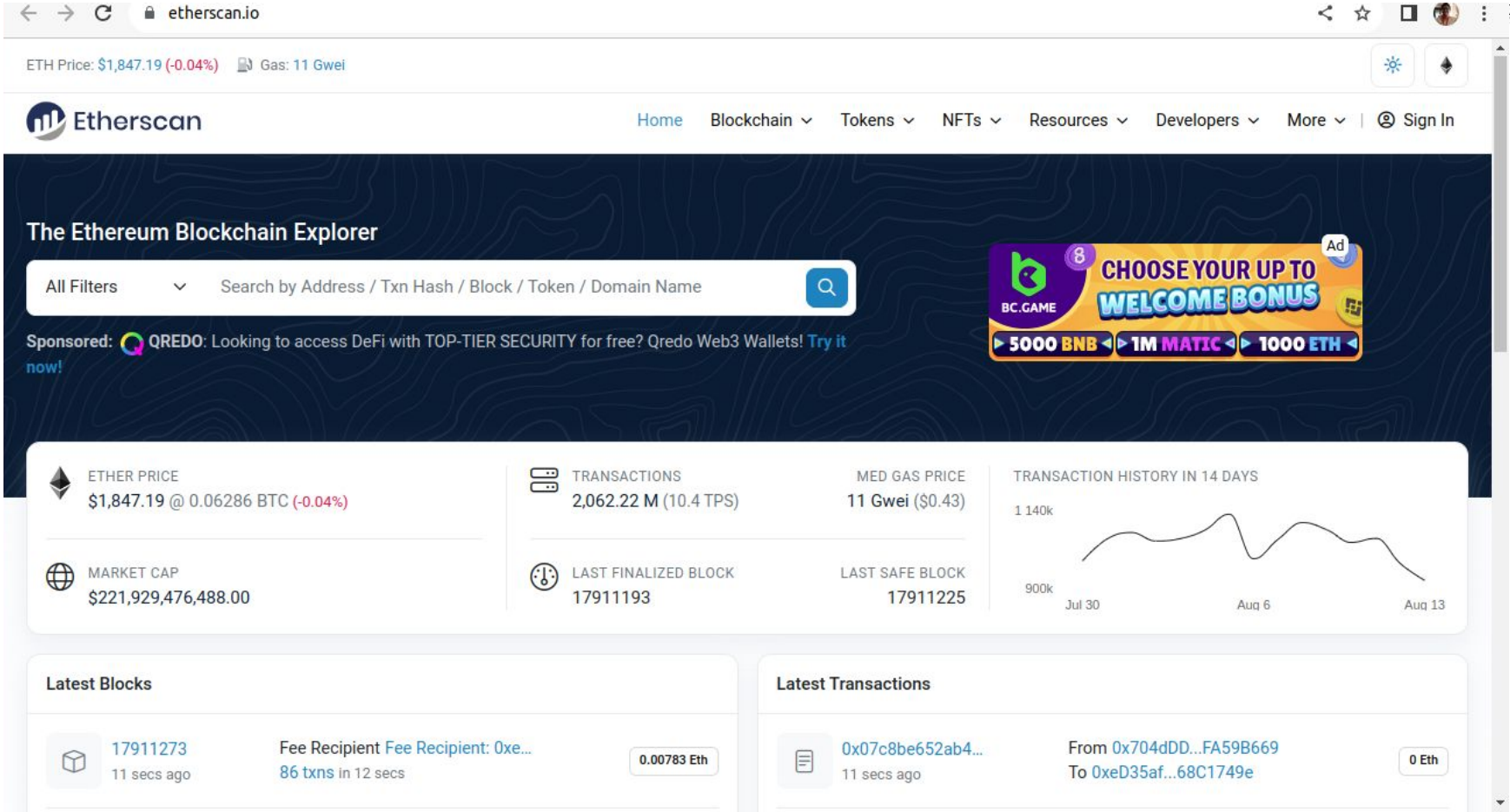


Case study of Ganache for Ethereum blockchain

On the Ganache, the Contract call is listed



Exploring etherscan.io



Courtesy : Etherscan.io

Exploring etherscan.io and ether block structure

What is Etherscan ?

- effective block explorer
- abstracts the complexity of retrieving data on the blockchain
- by providing a simple interface for Ethereum blockchain.
- allows users to look through the transactions, smart contracts, wallet addresses, blocks, and other on-chain data related to Ethereum-based trades and assets.
- all Ethereum interactions are publicly recorded on Blockchain.
- Helps to visualize past transactions on the Ethereum network that spans from transferring coins to purchasing non-fungible tokens (NFTs).

Exploring etherscan.io and ether block structure

Why Do You need Etherscan?

- allows users to **search and explore** the Ethereum blockchain
  - for transactions, tokens, addresses, and other activities.
- Provides **Token Tracker** for **Ether** by providing information,
  - such as price, holders, total supply, and transfers.
- It allows **to monitor high-profile NFT holders' portfolios.**
  - track what a certain NFT project or company is doing with its funds.
  - Eg : Gamestop dumped $42 million worth of NFX tokens, shortly after it received a $100 million grant from Immutable X. The proof of this incident was provided by Etherscan.
- Etherscan **helps the victims of crypto scams.**
  - By tracking the transaction of NFT projects' potential rug pulls and NFT scams,
- Etherscan **tracks the "whales" wallet which holds a large amount of ETH.**
  - helpful as you can predict the price movement with this information.

Courtesy : Sensorium

Exploring etherscan.io and ether block structure

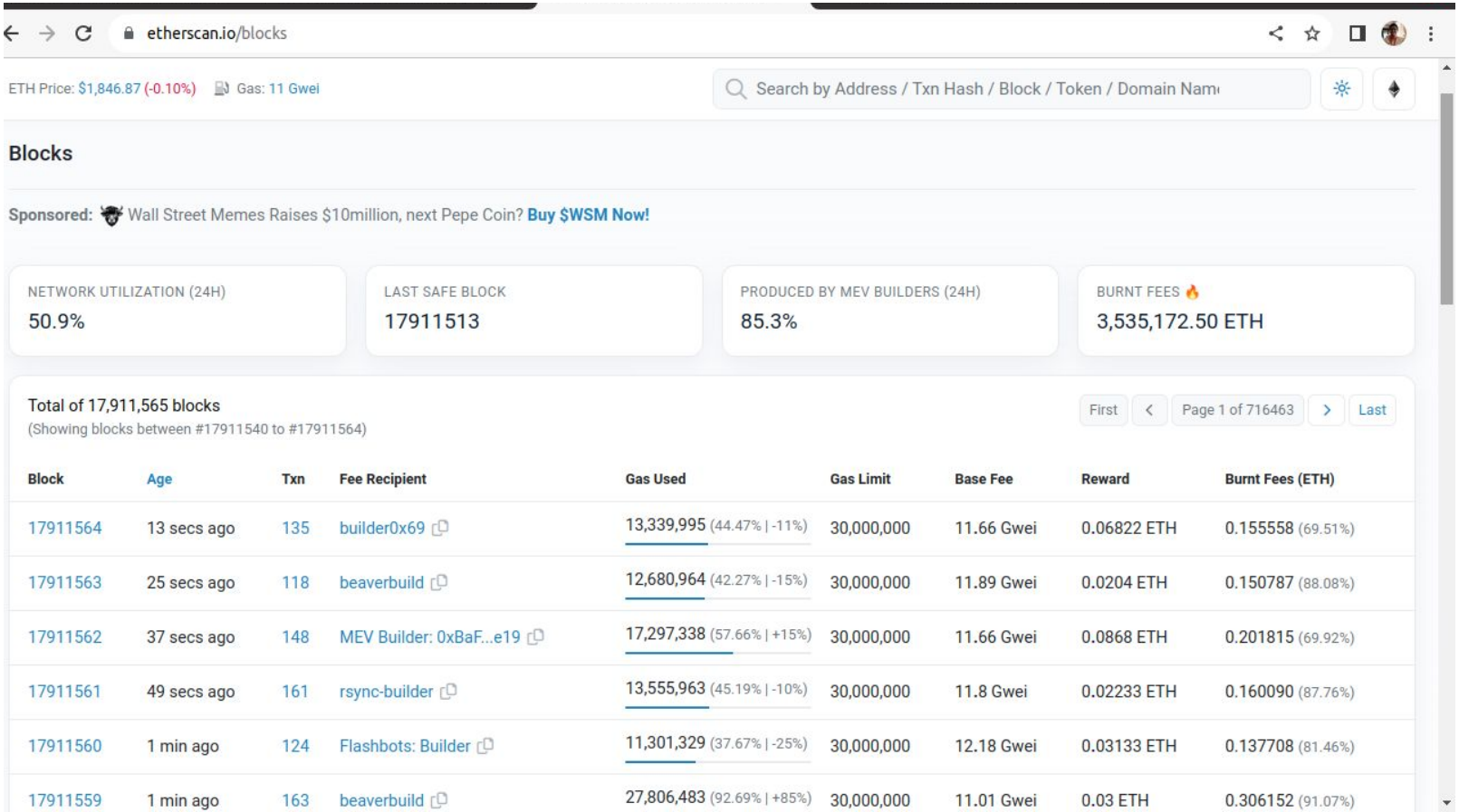
Why Do You need Etherscan?

- The use cases may involve:
  - Searching and viewing wallets and transactions
  - Tracking gas fees
  - Interacting with smart contracts
  - **Revoking or reviewing** your token approvals
  - Using **Token ignore list feature to hide your tokens**
  - Staying up-to-date with the Ethereum ecosystem

Note:

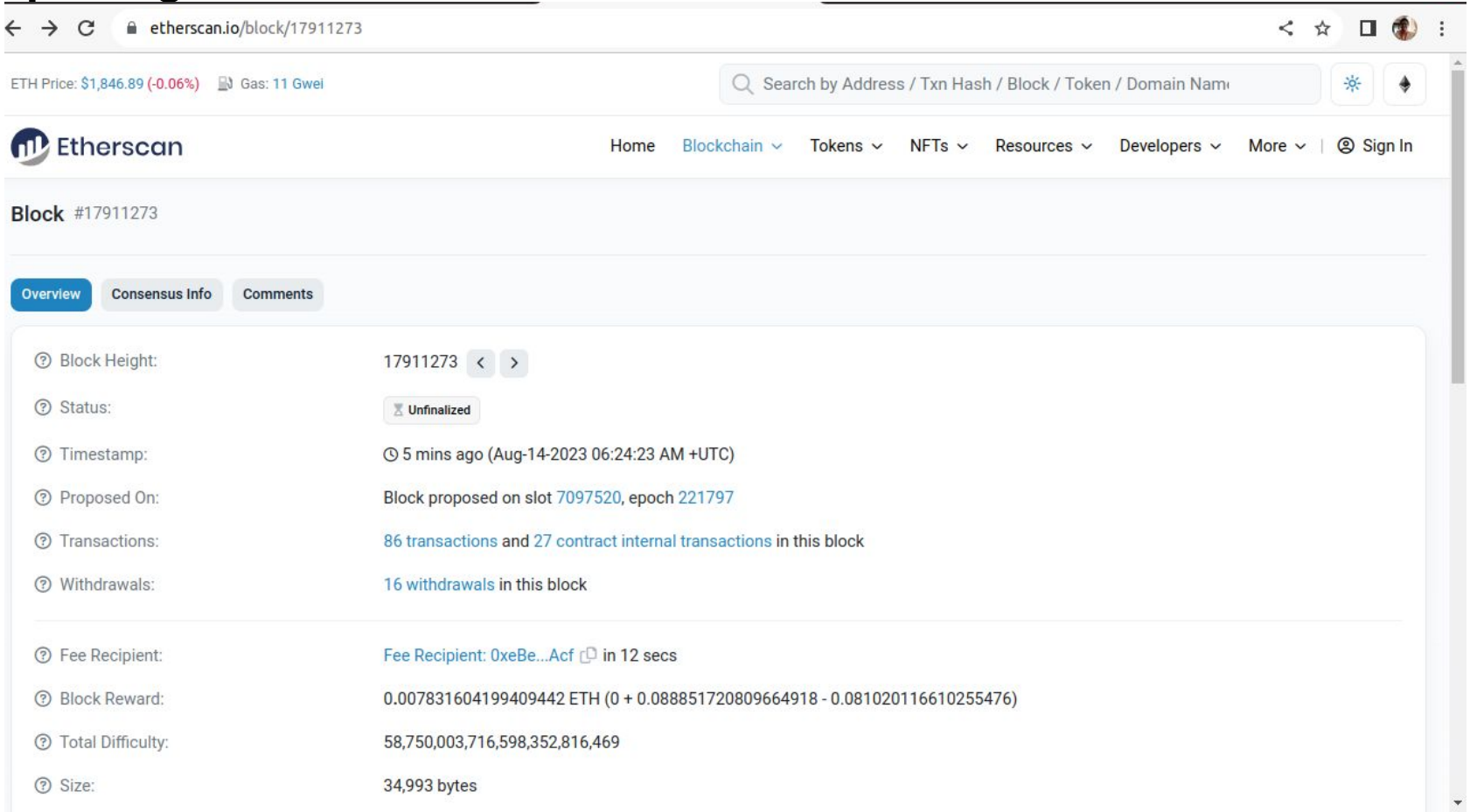
- **Etherscan is not a wallet;**
- Use Etherscan to search your Ethereum wallet and track the transactions
- If you want to **enter into a transaction or use ETH**, access your **crypto exchange or wallet.**

Exploring etherscan.io - Blocks



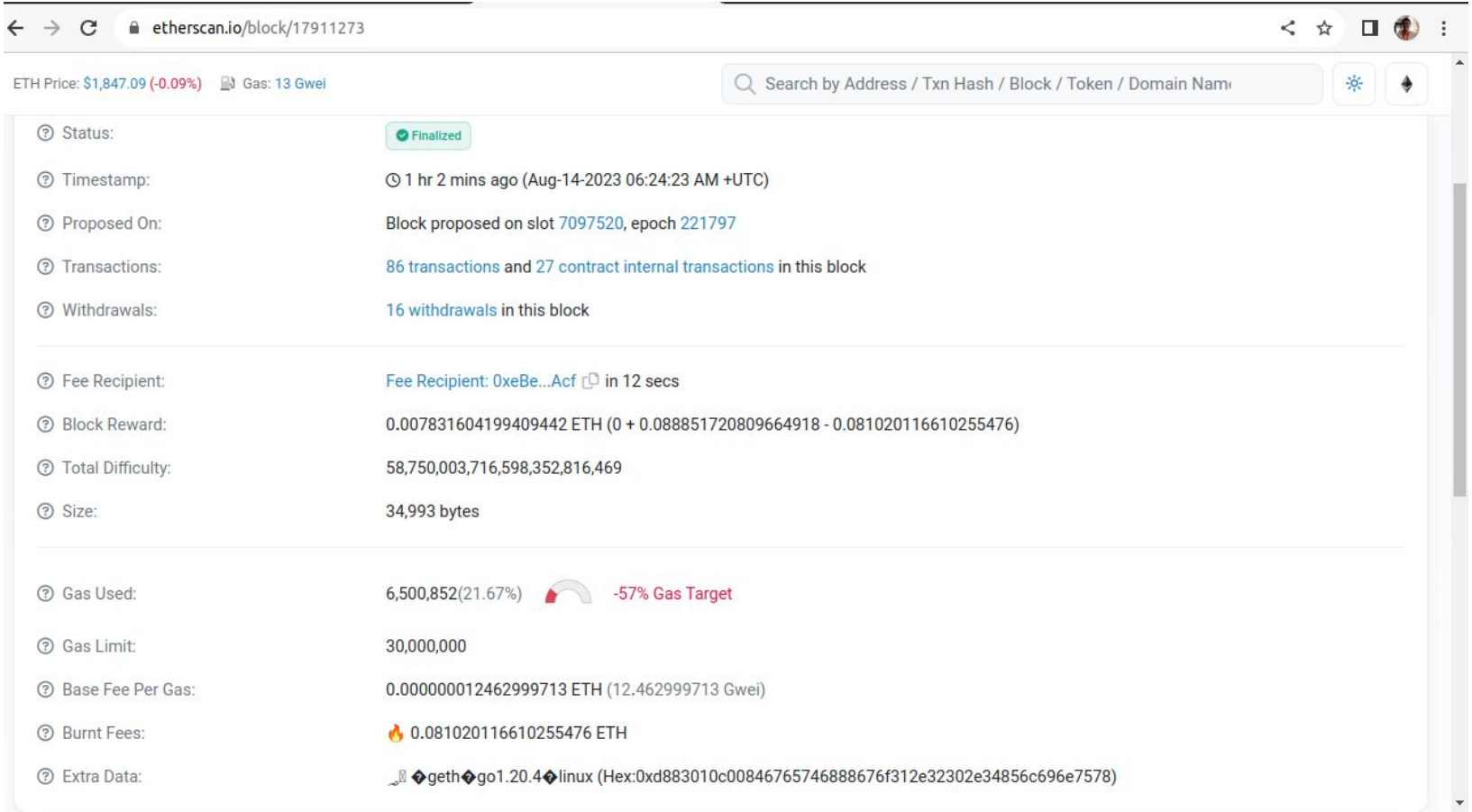
Courtesy : Etherscan.io

Exploring etherscan.io - ether block structure



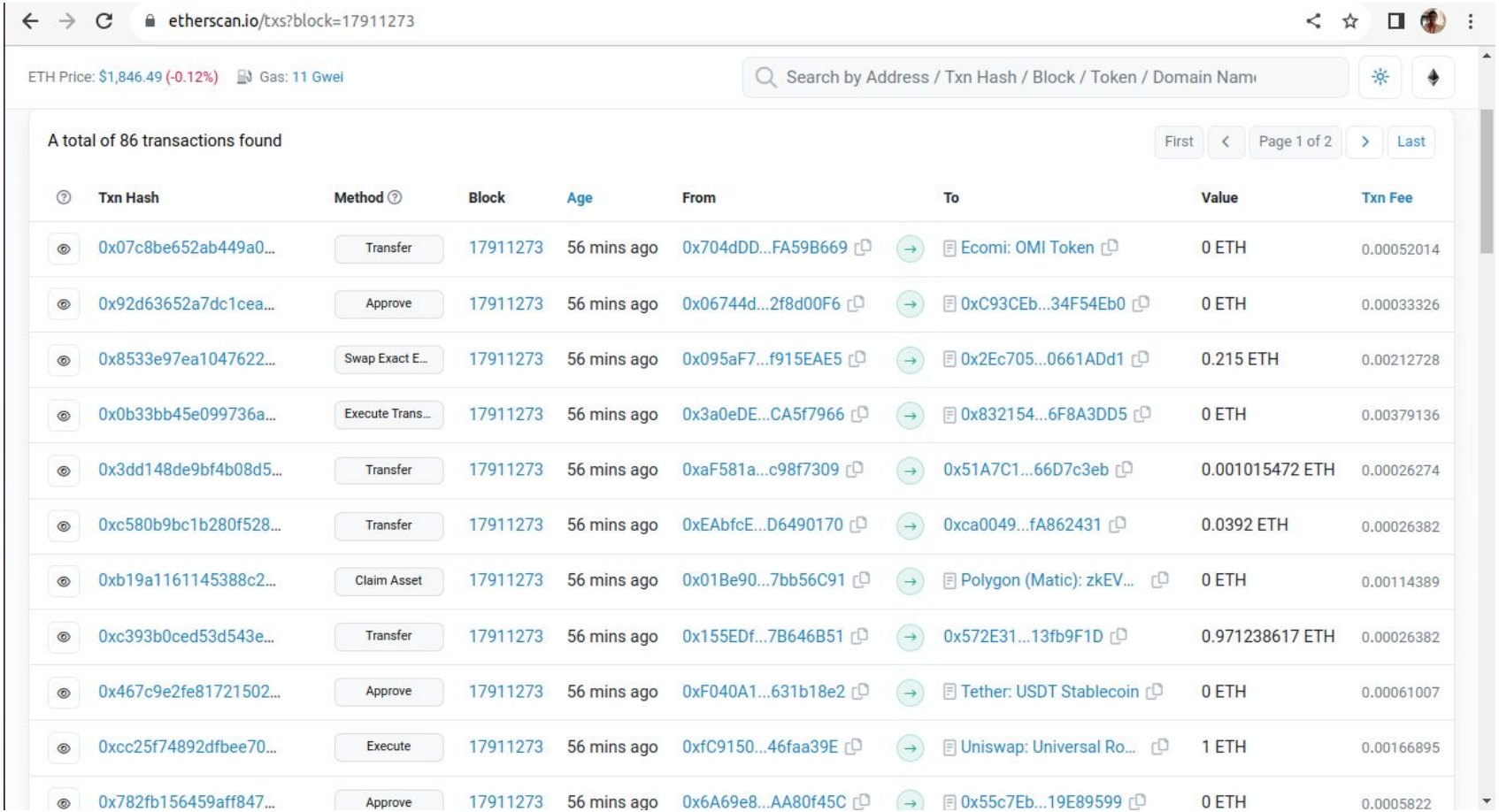
Courtesy : Etherscan.io

Exploring etherscan.io - ether block structure (after Finalized)



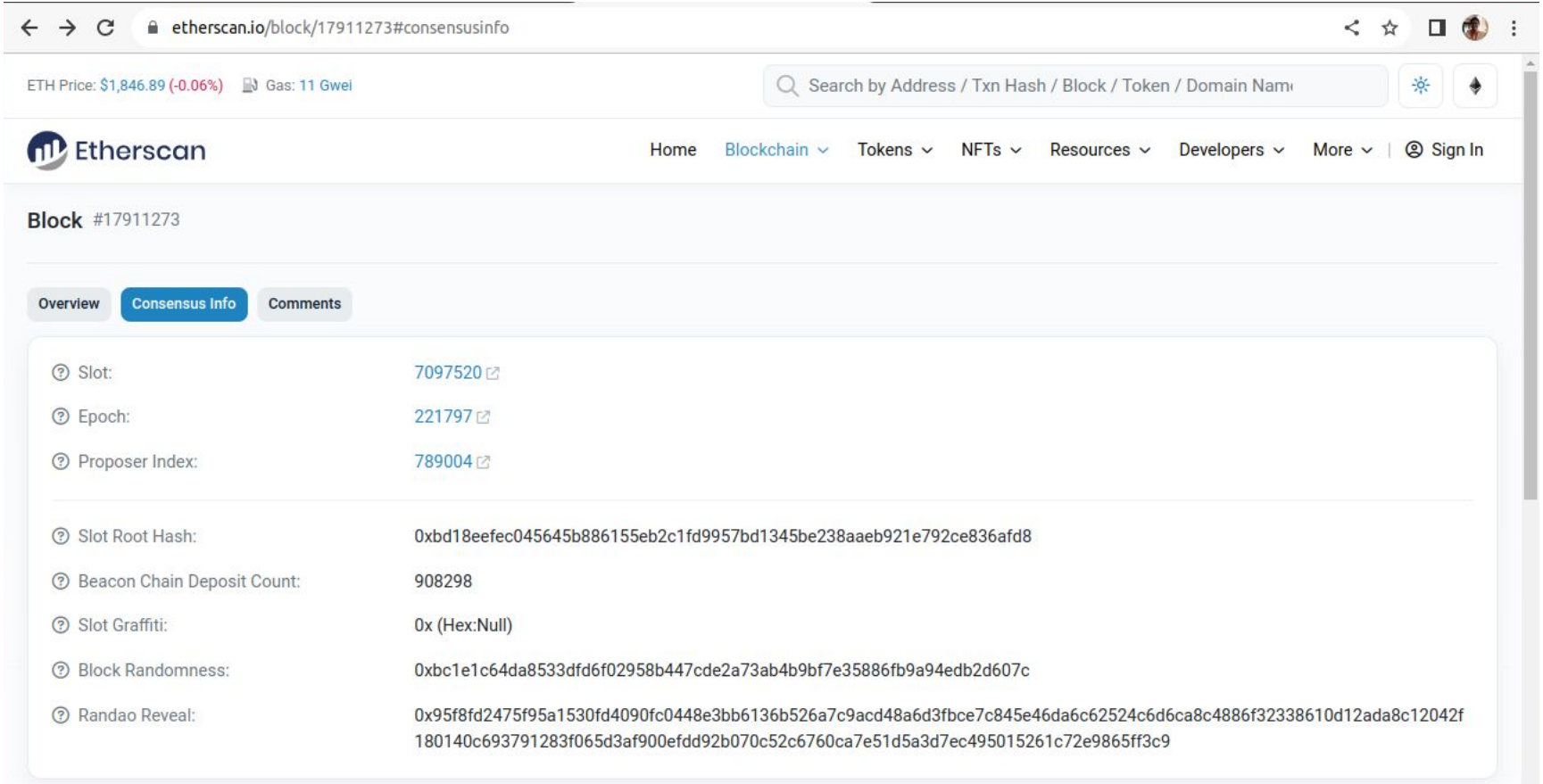
Courtesy : Etherscan.io

Exploring etherscan.io - ether block structure (Transactions)



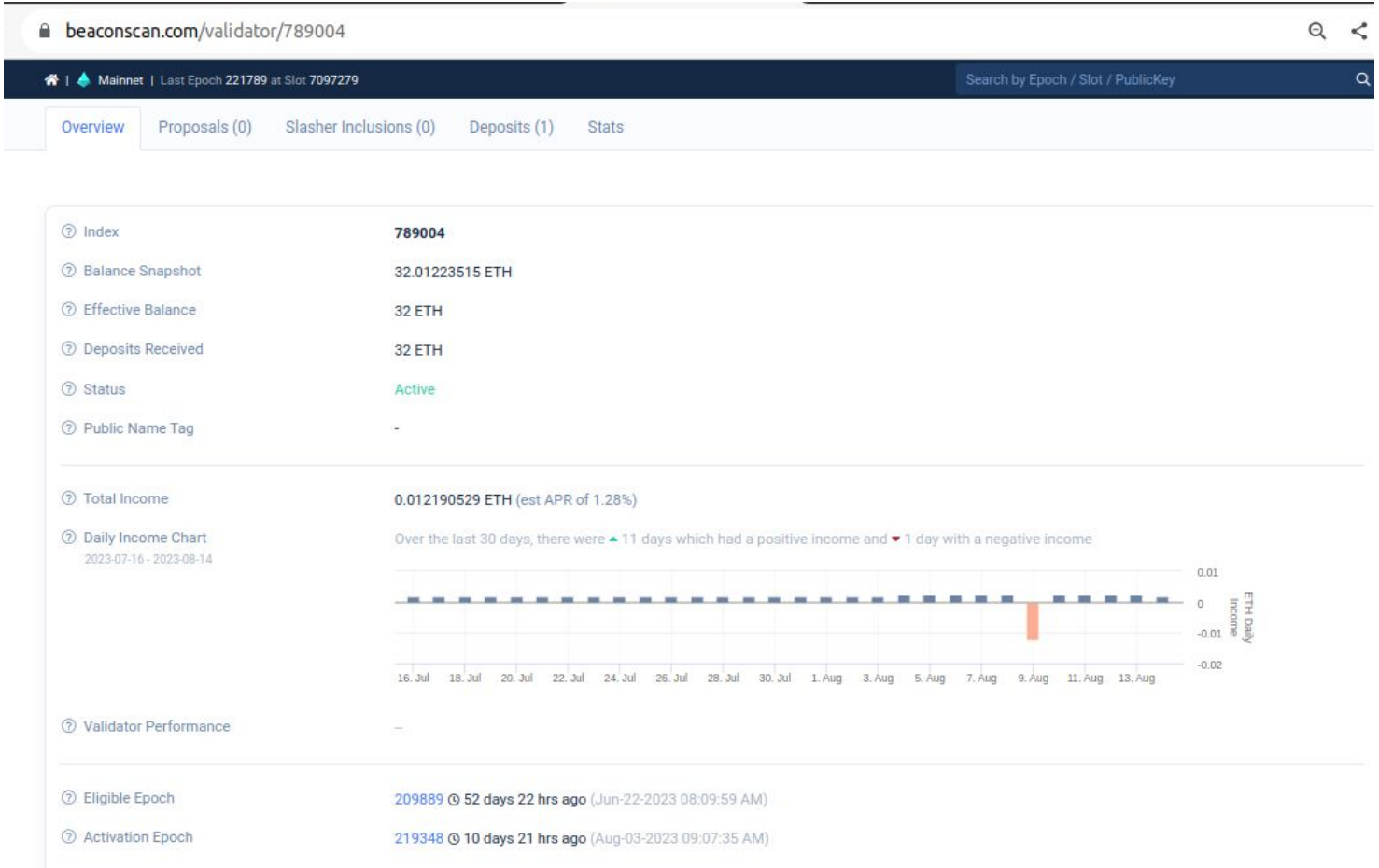
Courtesy : Etherscan.io

Exploring etherscan.io - ether block structure (Consensus Info)



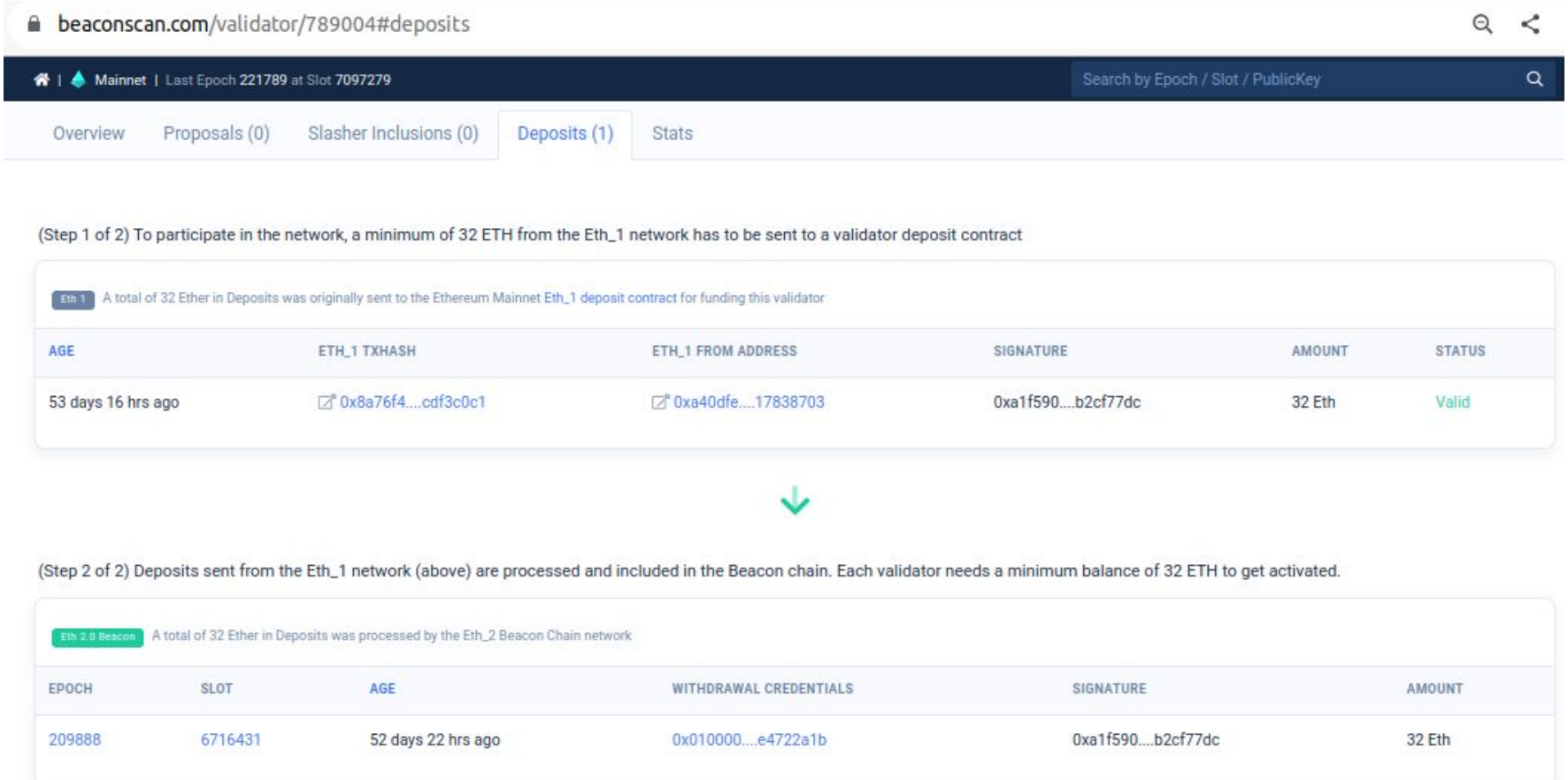
Courtesy : Etherscan.io

Exploring etherscan.io and ether block structure (Consensus Info - validator)



Courtesy : Etherscan.io

Exploring etherscan.io and ether block structure (Consensus Info - validator)



Courtesy : Etherscan.io

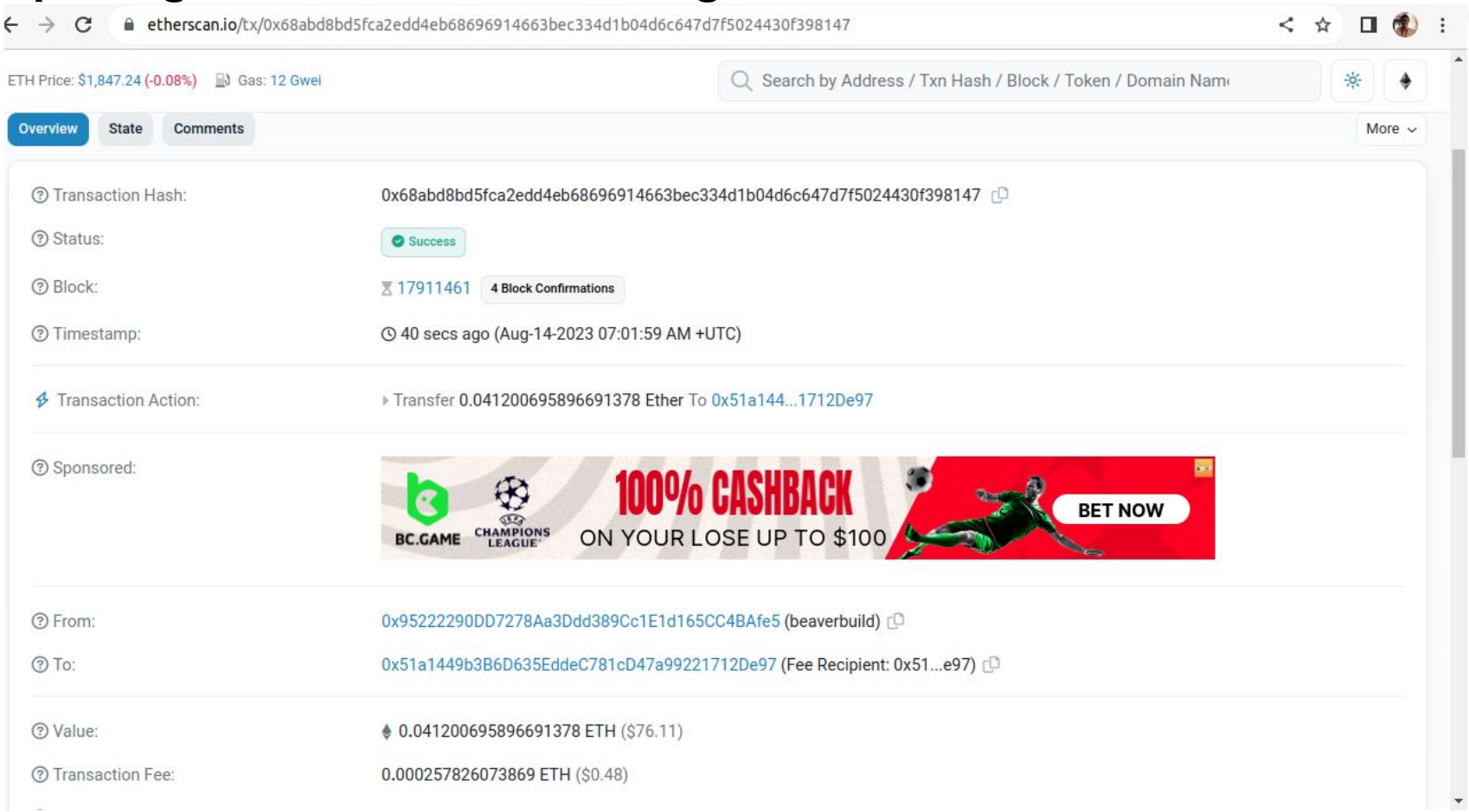
Exploring etherscan.io - Checking Transactions

- **Keeping track of Ethereum blockchain transactions (**most fundamental use case of Etherscan).
- If a user understands how to follow cryptocurrency, he will be able to unlock the rest of the cryptocurrency information.
- Steps to see the history of a particular crypto transaction on Ethereum
    1. Find the transaction hash(Tx Hash) or transaction ID (TxID).
       (identify a particular transaction on the blockchain.)
    2. Go to the Etherscan search bar and enter the TxID or Tx Hash and click “search”
    3. A page will display information related to that transaction, such as its status (successful, failed, pending), time and date of transaction, and the number of blocks in which the transaction was recorded.
    4. You can also find senders' and receivers’ addresses, gas prices, and transaction fees. Click “see more” to get these additional details.

Note :

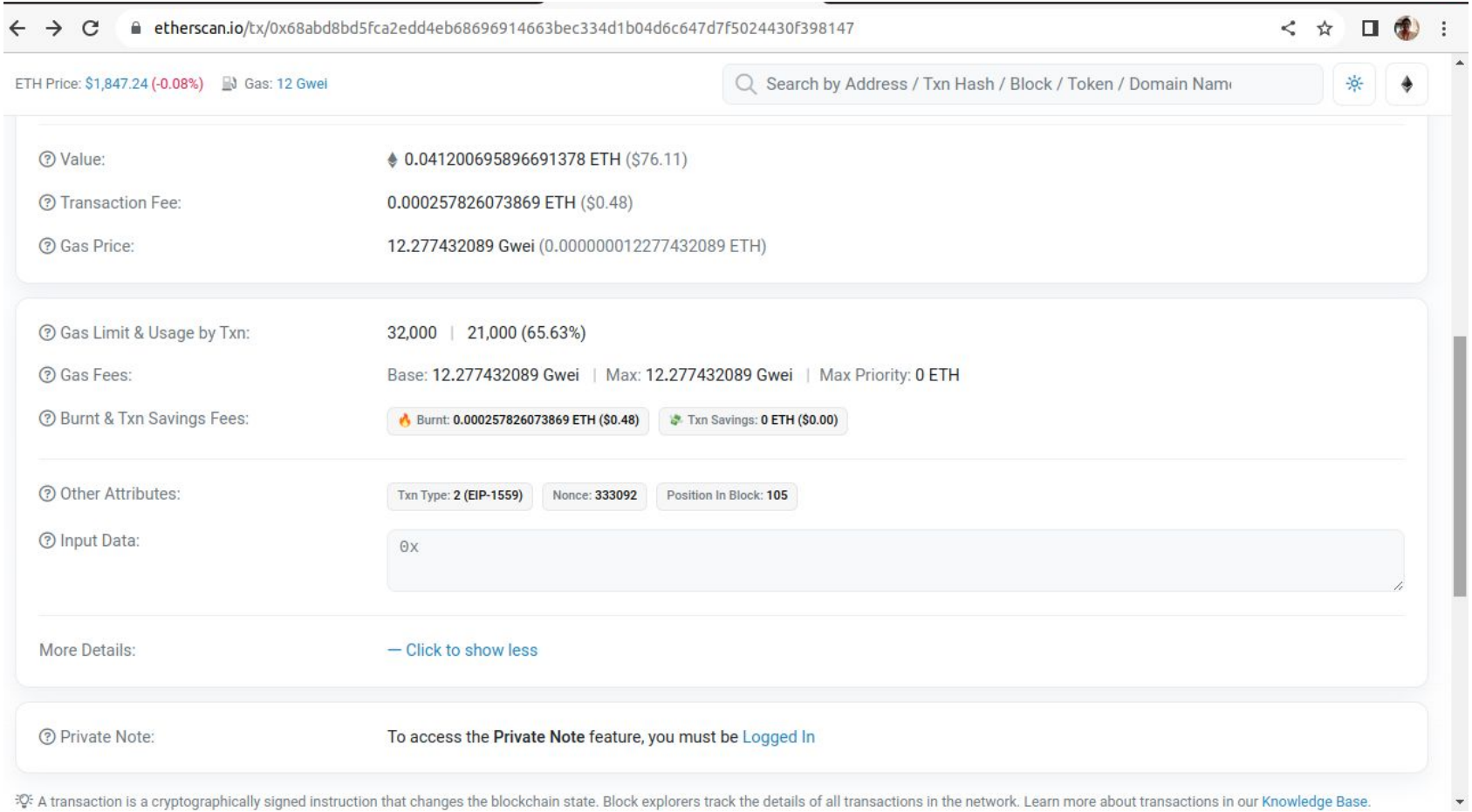
- One can also access the transaction using the recipient's wallet address.
- Insert the recipient’s wallet address in the search bar.
- All the transaction details associated with a particular wallet address will be displayed at the bottom of the page.

Exploring etherscan.io - Checking Transactions



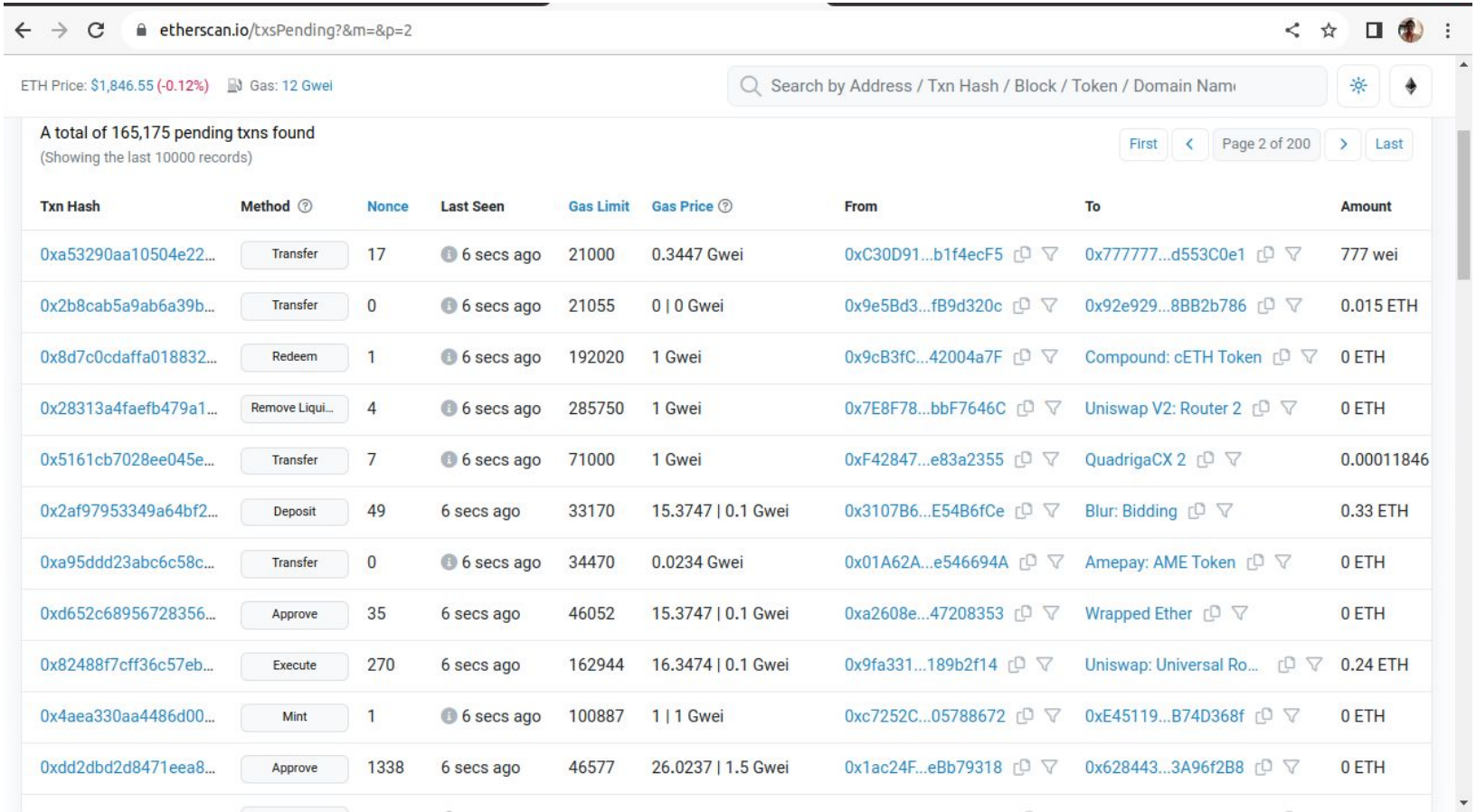
Courtesy : Etherscan.io - Tx

Exploring etherscan.io - Checking Transactions



Courtesy : Etherscan.io - Tx

Exploring etherscan.io - Checking Transactions (pending)



Txn Hash	Method	Nonce	Last Seen	Gas Limit	Gas Price	From	To	Amount
0xa53290aa10504e22...	Transfer	17	6 secs ago	21000	0.3447 Gwei	0xC30D91...b1f4ecF5	0x777777...d553C0e1	777 wei
0x2b8cab5a9ab6a39b...	Transfer	0	6 secs ago	21055	0 | 0 Gwei	0x9e5Bd3...fB9d320c	0x92e929...8BB2b786	0.015 ETH
0x8d7c0cdaffa018832...	Redeem	1	6 secs ago	192020	1 Gwei	0x9cB3fC...42004a7F	Compound: cETH Token	0 ETH
0x28313a4faefb479a1...	Remove Liqui...	4	6 secs ago	285750	1 Gwei	0x7E8F78...bbF7646C	Uniswap V2: Router 2	0 ETH
0x5161cb7028ee045e...	Transfer	7	6 secs ago	71000	1 Gwei	0xF42847...e83a2355	QuadrigaCX 2	0.00011846
0x2af97953349a64bf2...	Deposit	49	6 secs ago	33170	15.3747 | 0.1 Gwei	0x3107B6...E54B6fCe	Blur: Bidding	0.33 ETH
0xa95ddd23abc6c58c...	Transfer	0	6 secs ago	34470	0.0234 Gwei	0x01A62A...e546694A	Amepay: AME Token	0 ETH
0xd652c68956728356...	Approve	35	6 secs ago	46052	15.3747 | 0.1 Gwei	0xa2608e...47208353	Wrapped Ether	0 ETH
0x82488f7cff36c57eb...	Execute	270	6 secs ago	162944	16.3474 | 0.1 Gwei	0x9fa331...189b2f14	Uniswap: Universal Ro...	0.24 ETH
0x4aea330aa4486d00...	Mint	1	6 secs ago	100887	1 | 1 Gwei	0xc7252C...05788672	0xE45119...B74D368f	0 ETH
0xdd2dbd2d8471eea8...	Approve	1338	6 secs ago	46577	26.0237 | 1.5 Gwei	0x1ac24F...eBb79318	0x628443...3A96f2B8	0 ETH

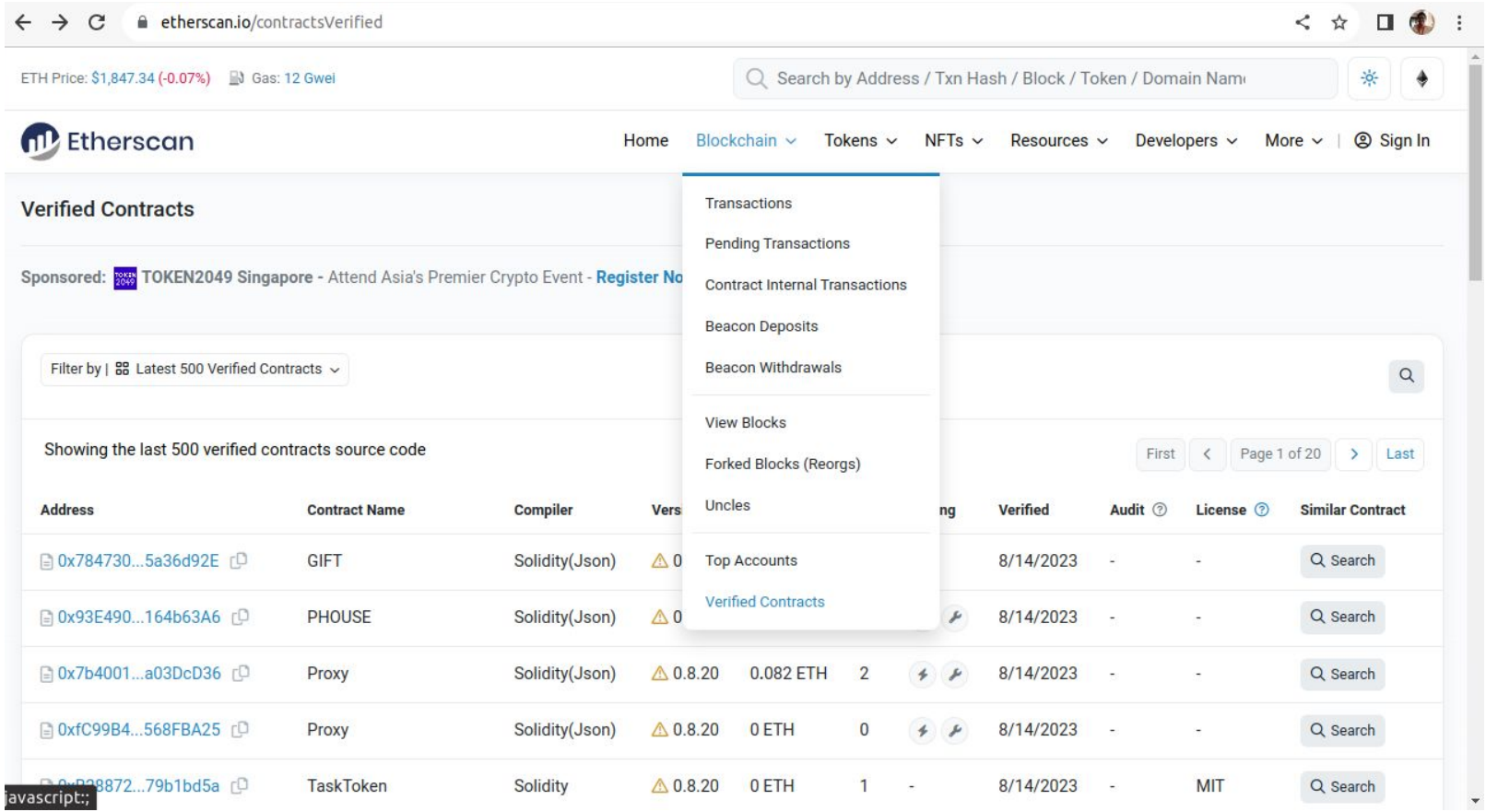
Courtesy : Etherscan.io - Tx

Exploring etherscan.io - Checking Smart Contracts

1. Helpful for **users who are regularly in contact with smart contracts** of decentralized applications.
   - They can confirm the transfer of funds to the right contract.
   - The **smart contract addresses** contain information, such as
     - token logic, how the token was transferred, and a “code” tab to access smart contracts’ source code.
   - To **access information related to smart contracts.**
     - Users need to **copy and paste the contract address** on the search icon
2. Allows users to read contracts and edit them.
   - Click on the “Read Contract” and “Write Contract” option on Etherscan.
3. **Minting NFTs** from smart contracts is another use case of Etherscan.
   - The verified contracts feature a checkmark.
   - **One will not be able to interact if the contract is not verified.**
   - If you have access to the application binary interface (ABI) of a contract, you can interact with an unverified contract as well.
4. Allows users to search for the contract.
   - You can see the code of contract, its balance, and transactions made to that contract.

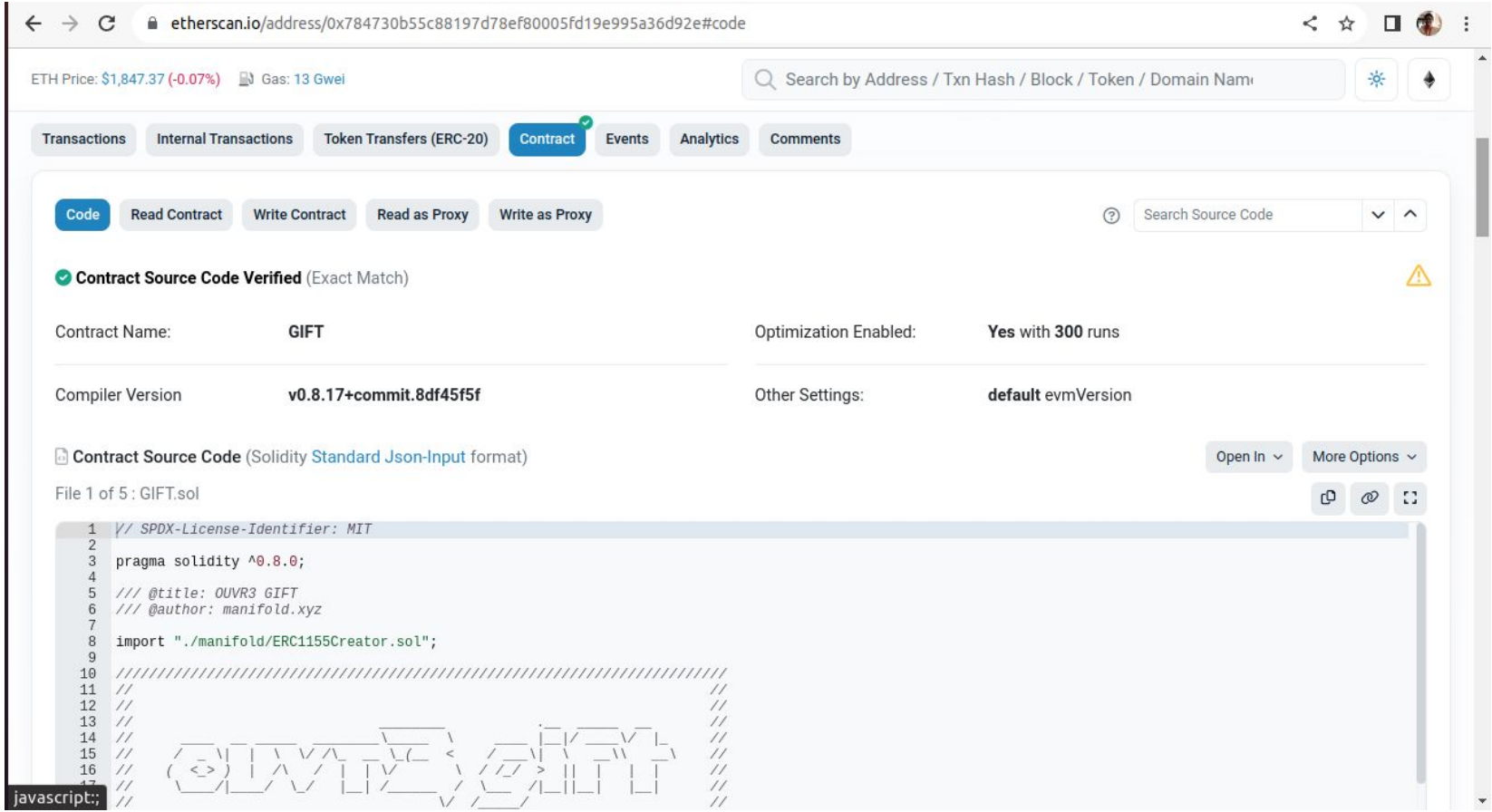
Courtesy : Sensorium

Exploring etherscan.io - Checking Smart Contracts



Courtesy : Etherscan.io - Verified Contracts

Exploring etherscan.io - Checking Smart Contracts



Courtesy : Etherscan.io - Verified Contract

Exploring etherscan.io - Checking Gas Prices

- "Gas" is a unit used in the Ethereum platform to **describe the amount of computational power needed to execute the transaction.**
- One need to pay a certain amount to perform a certain transaction on Ethereum.
- This amount is variable as it depends on the network traffic and the number of blocks included in the transaction.

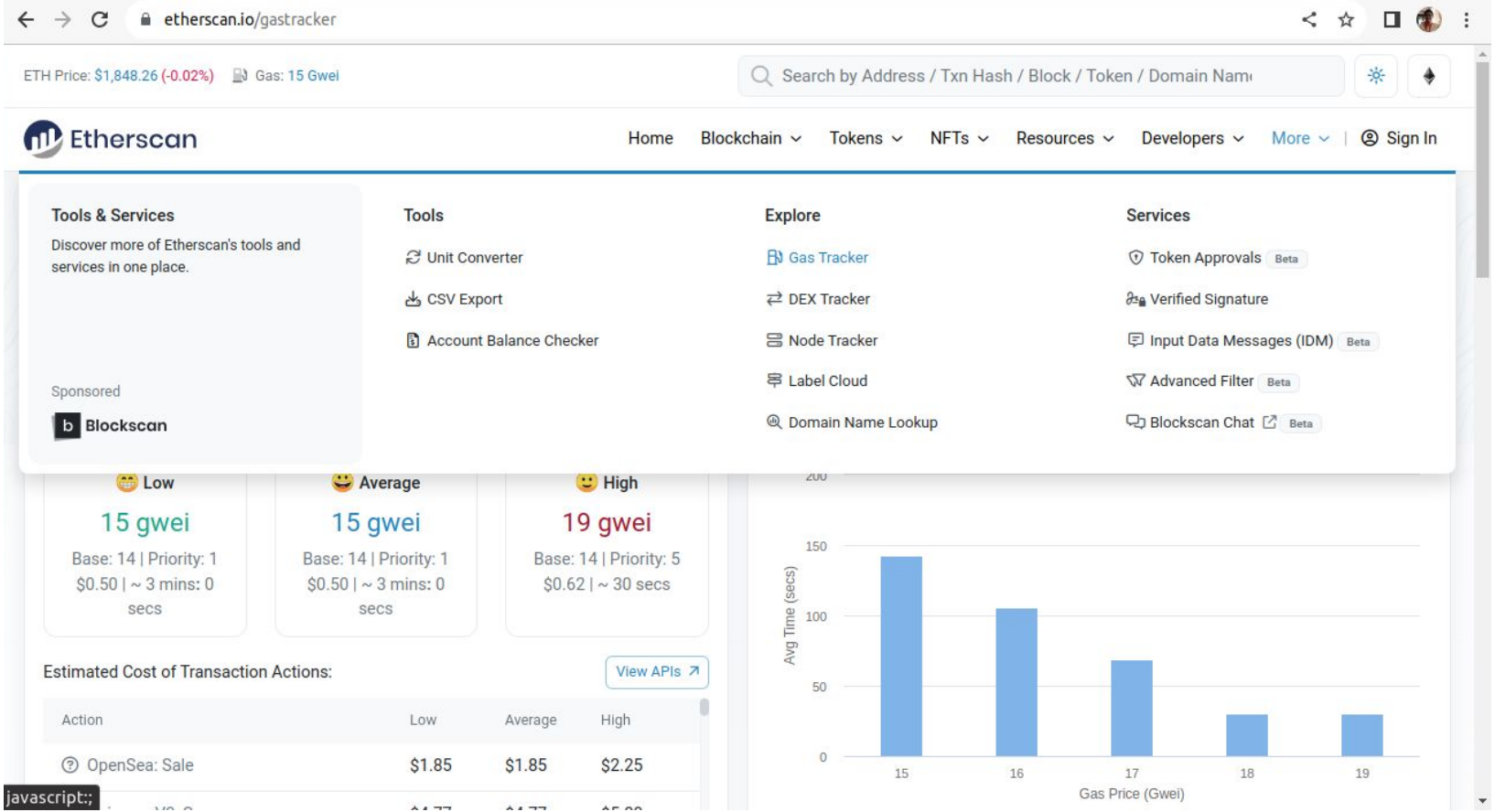
Etherscan offers a **dedicated page to track the gas fees of a specific transaction**.

- This page shows the average gas fee for the last seven days and the last few blocks.
- This can help you decide the best time for making a transaction.
- The **"gas tracker"** feature shows the difference in time and price at different gas prices.

Etherscan also roughly **predicts the gas fee based on the amount to be transferred and network traffic.**

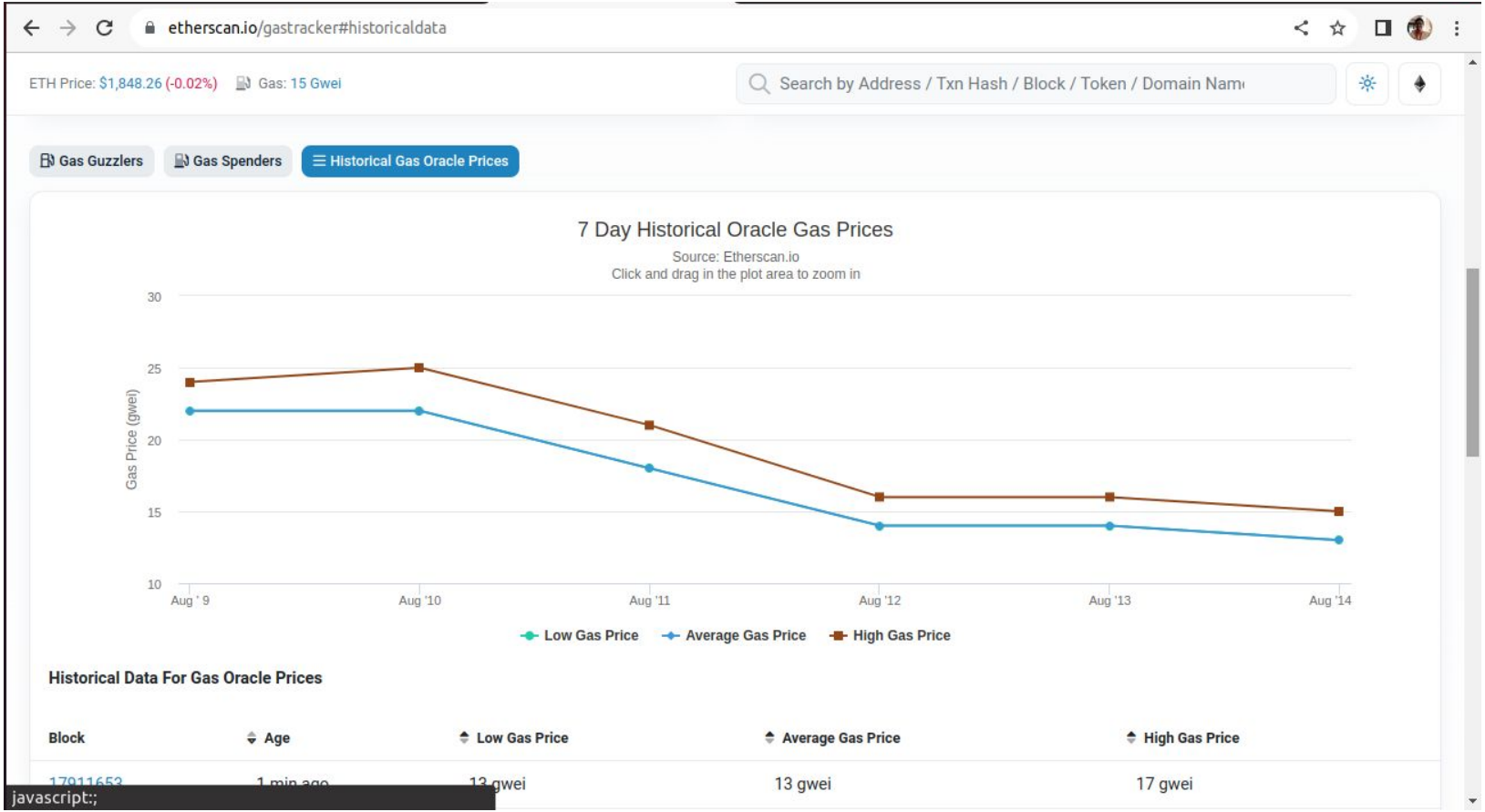
- You will find the average gas fee on Etherscan.
- Below that you will get an estimated cost of the transaction.
- **Higher gas prices** reflect the higher speed of the transactions. .

Exploring etherscan.io - Checking Gas Prices



Courtesy : Etherscan.io - Gas Tracker

Exploring etherscan.io - Checking Gas Prices



Courtesy : Etherscan.io - Gas Tracker

Exploring etherscan.io - Checking NFTs

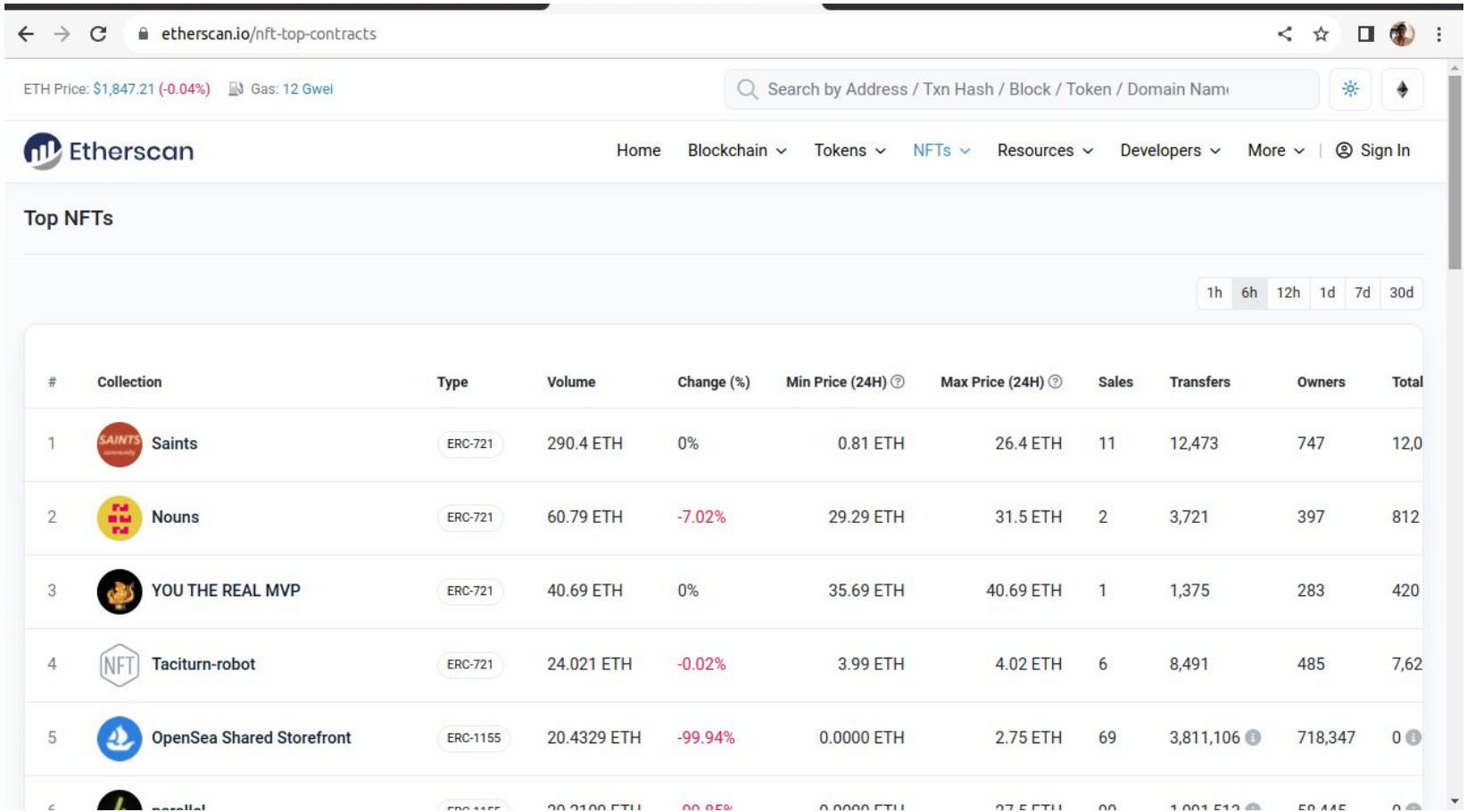
- Feature launched in 2021 by which you can directly check your NFTs.
- This **allows users to buy and sell their digital collectibles increasing transparency on transactions and ownership**.

Steps for checking NFTs on Etherscan:

1. Click the Etherscan homepage, and find the search bar at the top of the page.
2. Insert your wallet address in the search icon and hit “Search”.
3. In the mid-screen, click ‘Erc721 Token Txns’ to see a list of all NFTs associated with the given wallet address.
4. In order to access additional details, go to the detailed column and click “View NFTs”.

Courtesy : Sensorium

Exploring etherscan.io - top NFT's



#	Collection	Type	Volume	Change (%)	Min Price (24H)	Max Price (24H)	Sales	Transfers	Owners	Total
1	Saints	ERC-721	290.4 ETH	0%	0.81 ETH	26.4 ETH	11	12,473	747	12,0
2	Nouns	ERC-721	60.79 ETH	-7.02%	29.29 ETH	31.5 ETH	2	3,721	397	812
3	YOU THE REAL MVP	ERC-721	40.69 ETH	0%	35.69 ETH	40.69 ETH	1	1,375	283	420
4	Taciturn-robot	ERC-721	24.021 ETH	-0.02%	3.99 ETH	4.02 ETH	6	8,491	485	7,62
5	OpenSea Shared Storefront	ERC-1155	20.4329 ETH	-99.94%	0.0000 ETH	2.75 ETH	69	3,811,106	718,347	0

Courtesy : Etherscan.io

Exploring etherscan.io - Working

- Etherscan serves as a search engine for ethereum.
- Analogy : consider it as a Google but for the cryptocurrency world.
- In fact, these block explorers are called **“blockchain browsers” or “blockchain search engines”**.

Note :

- One can access wallet, transactions, and smart contract
- Can also identify any suspicious activities.
- However, **Etherscan is not a wallet**,
  - it is a source of information that allows users to track their transactions on the Ethereum network, index this information, and add it to its website.

Etherscan

- **combines different technologies, such as blockchain explorers, decentralized applications (Dapps), and Web3 libraries**,
- **displays the information in an easy-to-access format.**

Courtesy : Sensorium

Exploring etherscan.io - Working

The working of Etherscan is based on three parts:

1. **Data Retrieval:**
   - Etherscan uses an Application Programming Interface (API) to access data.
   - APIs allow one software to interact with another.
   - API connects block explorers to the Ethereum blockchain.
   - Etherscan **uses JSON-RPC 2.0 specification**,
     - which means it utilizes JSON (JavaScript Object Notation) t**o communicate with Ethereum.**
     - These APIs provide users access to the transactions and blocks on Ethereum.
2. **Data Storage:**
   - Etherscan stores the data in the form of relational tables in the “SQL Database”.
   - The relational format means the data in one table is related to the data in other tables.
   - This relationship helps explorers to display information easily.
3. **Data Production:**
   - Data displayed in the Database is converted to a human-understandable language.
   - Etherscan takes the help of different programming languages, such as HTML, CSS, and JavaScript.
     - **HTML** structures the data on block explorers.
     - **CSS** gives styles to the content.
     - **JavaScript** helps data to interact with the user.

Courtesy : Sensorium

Exploring etherscan.io - General Questions

Is Etherscan free?

- Yes! Etherscan is a free tool that allows people to check their wallet and transaction status without any service charges.
- It does not store or hold your cryptocurrency as it is not a cryptocurrency wallet service provider.
- It just offers an interface to view and track your past transactions.

Do you need an account to use Etherscan?

- No! You do not need to create an account to access your ether transactions using Etherscan.
- However, you can open an account to access certain Etherscan features.
- For instance, If you want to build data streams,
  - utilize toolchains, and
  - set notifications to alert income or expenditure.
- These **add-ons are only available if you open an account on Etherscan**.
- This search engine is **managed autonomously by dedicated individuals to make decentralized information and technical apps feasible.**

Courtesy : Sensorium

Exploring etherscan.io - Live Demo (Google CoLab)

Extracting Live data with API Keys

- Go to https://etherscan.io/
- Sign Up - create an account
- Verify your account from email
- Login to your account
- Go to API Keys
- Create App by giving name
- Copy the API Key generated into the code

```python
import requests

def get_latest_block(api_key):
    url = "https://api.etherscan.io/api"
    params = {
        "module": "proxy",
        "action": "eth_getBlockByNumber",
        "tag": "latest",
        "boolean": "true",
        "apikey": api_key,
    }

    try:
        response = requests.get(url, params=params)
        if response.status_code == 200:
            data = response.json()
            return data["result"]
        else:
            print("Request failed with status code:", response.status_code)
    except requests.RequestException as e:
        print("Request failed:", str(e))

    return None

# Replace "YOUR_API_KEY" with your actual API key
api_key = "EXK36NERHPD5487RG5F2RQI669A79VQUHR"            # Paste your API Key
```

Exploring etherscan.io - Live Demo (Google CoLab)

Extracting Live data with API Keys

- Go to https://etherscan.io/
- Sign Up - create an account
- Verify your account from email
- Login to your account
- Go to API Keys
- Create App by giving name
- Copy the API Key generated into the code

```
print ("Current date and time : ", now.strftime("%d-%B-%Y"))
latest_block = get_latest_block(api_key)
#print(latest_block)
if latest_block is not None:
    print("Latest block information:")
    print("Block Number:", int(latest_block["number"], 16))
    print("Timestamp:", int(latest_block["timestamp"], 16))
    print("Miner Address:", latest_block["miner"])
    print("Difficulty:", int(latest_block["difficulty"], 16))
    print("Total Difficulty:", int(latest_block["totalDifficulty"], 16))
    print("Gas Limit:", int(latest_block["gasLimit"], 16))
    print("Gas Used:", int(latest_block["gasUsed"], 16))
    print("Transaction Count:", len(latest_block["transactions"]))
    print("Transactions:", latest_block["transactions"])
    # Add more fields as per your requirements
    # print("Transactions:")
    # for tx in latest_block["transactions"]:
    #     print(tx)
```

```
Current date and time :  13-August-2023
Latest block information:
Block Number: 17902335
Timestamp: 1691886227
Miner Address: 0x1f9090aae28b8a3dceadf281b0f12828e676c326
Difficulty: 0
Total Difficulty: 58750003716598352816469
Gas Limit: 30000000
Gas Used: 12180476
Transaction Count: 123
```

Exploring infura.io - Live Demo (Google CoLab)

```python
import requests

# Infura HTTP endpoint
infura_url = 'https://mainnet.infura.io/v3/7329beded7a74ad085a6144b63645314'

# Make a request to retrieve the latest block number
response = requests.post(
    infura_url,
    json={
        "jsonrpc": "2.0",
        "method": "eth_blockNumber",
        "params": [],
        "id": 1
    }
)
if response.status_code == 200:
    result = response.json()
    latest_block_number = int(result["result"], 16)  # Convert hexadecimal to decimal

    print("Latest Block Number:",latest_block_number)
    # Print the desired information
    print("Miner Address:", latest_block["miner"]) # Error
    print("Difficulty:", int(latest_block["difficulty"], 16))
    print("Total Difficulty:", int(latest_block["totalDifficulty"], 16))
    print("Gas Limit:", int(latest_block["gasLimit"], 16))
    print("Gas Used:", int(latest_block["gasUsed"], 16))
    print("Transaction Count:", len(latest_block["transactions"]))
```

For extracting Live data,

Infura.io - Getting Started

```
Current date and time :  14-August-2023
Latest Block Number: 17911780
Miner Address: 0x690b9a9e9aa1c9db991c7721a92d351db4fac990
Difficulty: 0
Total Difficulty: 58750003716598352816469
Gas Limit: 29970705
Gas Used: 13555004
Transaction Count: 138
```