**Blockchain Documentation**

**Blockchain**

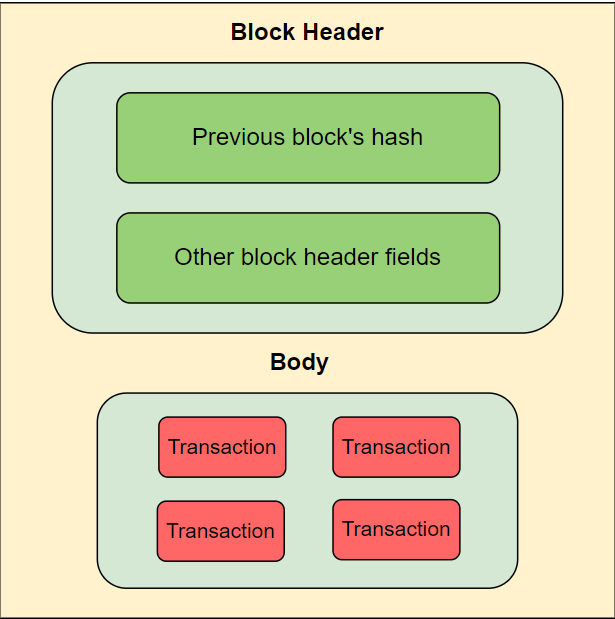
* Blockchain is a decentralized, distributed ledger technology that records transactions across multiple computers in a secure, transparent, and immutable manner.
* Each transaction is grouped into a “Block” and linked to the previous blocks.
* This chain is maintained by a network of nodes.
* Blockchain technology is most associated with cryptocurrencies like bitcoin, but its application extends to various industries, including finance, supply chain management, healthcare, and more.
* Blockchain is a type of shared database that differs from a typical database in the way it stores information; blockchains store data in blocks linked together via cryptography.
* Different types of information can be stored on a blockchain, but the most common use for transactions has been the ledger.
* In Bitcoin’s case, blockchain is decentralized so that no single person or group has control instead, all users collectively retain control.
* Decentralized blockchains are immutable which means that the data entry is irreversible. For Bitcoin, transactions are permanently recorded and viewable to anyone.
* **E.g.** Imagine you have a notebook where you write down all your expenses. Now, instead of keeping this notebook with you, you give a copy to all your friends. Whenever you spend money or receive some, everyone updates their copy of the notebook simultaneously.
* Now, imagine this notebook is digital, and instead of friends, there are thousands of computers all over the world. Also, the notebook isn't just about your expenses; it records transactions for everyone in a way that nobody can cheat. That's kind of how a blockchain works!

**How Does a Blockchain Work?**

* It starts with someone wanting to make a transaction. This could be anything from sending cryptocurrency to someone else or recording some kind of data like a contract.
* Once the transaction is requested, it’s broadcasted to a network of computers, called nodes, connected through the internet. These nodes verify the transaction to make sure it’s valid and not fraudulent.
* Verified transactions are grouped into a “block”. Think of it like a page in a ledger. This block contains a bunch of transactions, and it’s given a unique code called a hash.
* Each new block is then linked to a previous one, forming a chain of blocks hence the name “blockchain”. This linking is done using the unique hash code of the previous block.
* Before adding a new block to the chain, the nodes in the network need to agree that the transactions are valid. This agreement is called consensus, and it’s often achieved through mechanisms like proof of work or proof of stake.
* Once the consensus is reached, the new block is added to the chain, and the transaction is completed. Now, every node in the network updates its copy of the blockchain to include the new block.
* Once a block is added to the blockchain, it’s extremely difficult to alter because changing anything in one block would require changing every single block after it, which would need the consensus of the majority of the network. This immutability ensures the integrity and security of the blockchain.
* So, in simple terms, blockchain is like a digital ledger where transactions are verified, grouped into blocks, linked together, and then added to a chain of blocks, creating a secure and transparent record of transactions that can’t easily be changed.

**What is a Block?**

* A block is a place in a blockchain where information is stored and encrypted.
* Blocks are identified by long numbers that include encrypted transaction information from previous blocks and new transaction information.
* Blocks and the information within them must be verified by a network before new blocks can be created.
* In the case of cryptocurrency blockchains, the data stored in a block are transactions.
* These blocks are chained together by adding the previous block’s hash to the next block’s header.
* It keeps the order of the blocks intact and makes the data in the blocks immutable,



**Consensus:**

* Consensus in blockchain refers to the mechanism by which agreement is reached among network participants on the validity of transactions and the state of the blockchain.

OR

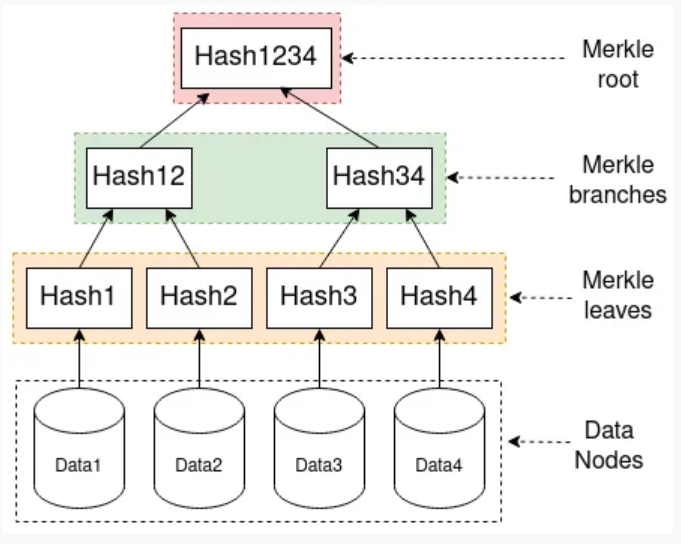
* Before adding a new block to the chain, the nodes in the network need to agree that the transactions are valid. This agreement is called consensus.
* Since blockchain operates in a decentralized manner, where there is no central authority to validate transactions, consensus protocols are essential for maintaining the integrity and security of the network.

Consensus Mechanisms:

1. Proof of Work (POW):
   * Minors complete solving complex mathematical puzzles to validate and add new blocks to the blockchain.
   * The first minor to solve the puzzle broadcasts the new block to the network, and other nodes verify the validity of the solution.
   * This process requires a significant amount of computational power and energy, but it ensures that the majority of the network agrees on the state of the blockchain.
2. Proof of Stake (POS):
   * Validators are chosen to create new blocks based on the amount of cryptocurrency they hold and are willing to “stake” as collateral.
   * Validators are selected through a deterministic process that considers factors such as the amount of cryptocurrency they hold and how long they’ve held it.
   * POS is considered more energy efficient than POW since it doesn’t require intensive computational tasks.
3. Delegated Proof of Stake (DPoS):
   * It is a variation of the POS consensus mechanism where token holders vote for a select number of delegates who are responsible for validating transactions and adding new blocks to the blockchain.
   * These delegates are typically chosen based on their reputation, performance, and contributions to the network.
   * DPoS aims to improve scalability and efficiency by delegating block validations to a smaller group of trusted entities.
4. Proof of Authority (PoA):
   * A predefined set of validators is authorized to create new blocks and validate transactions.
   * Validators are typically known and trusted entities within the network, such as companies or organizations.
   * PoA is often used in private or consortium blockchains where trust among participants is high and scalability is a priority.

**Merkle Root Tree:**

* A Merkle tree, also known as a binary hash tree or hash tree, is a fundamental data structure used in blockchain technology to efficiently store and verify the integrity of large sets of data, such as transactions in a block.
* Named after the innovator Ralph Merkle.
* A Merkle tree is constructed by repeatedly hashing a pair of data until a single hash, known as the Merkle root, is obtained.
* Work:
  + Data organization: In a blockchain, transactions are grouped into blocks. Instead of storing each transaction individually, which could be inefficient, the transactions are organized into a Merkle tree structure.
  + Hashing: Each transaction is hashed using a cryptographic hash function, such as SHA-256. This produced a fixed-size hash value that uniquely represents the transaction data.
  + Pairing and Hashing Again: The hash values of individual transactions are then paired together, and the pair is hashed again. This process continues until there is only one hash value left, known as the Merkle root.
  + Merkle Root: The Merkle root is the topmost hash value in a tree. It represents a summary of all the transactions in the block. Because of the properties of cryptographic hash functions, any change in the underlying transaction data would result in a completely different Merkle root.
  + Efficient Verification: When a block is transmitted across the networks, it includes the Merkle root along with other block information. Nodes in the network can then use the Merkle root to efficiently verify that the transaction in the block hasn’t been tampered with. They do this by requesting only the specific transaction data needed to compute the Merkle root, rather than the entire block.



**Hashing:**

* Hashing is a fundamental cryptographic process used to convert input data of any size into a fixed-size string of characters, which is typically a sequence of letters and numbers. This fixed-size string is known as a hash value or hash code.

**Smart Contracts:**

* A smart contract is a digital agreement signed and stored on a blockchain network that executes automatically when the contract’s terms and conditions are met.
* The T&C is written in blockchain-specific programming languages like solidity.
* Types:
  + Smart legal contract.
  + Decentralized autonomous organizations.
  + Application logic contracts.

**Ethereum:**

* Ethereum is one of the most well-known and widely used blockchain platforms, known for its smart contract functionality, Decentralized applications (DApps), and native cryptocurrency called Ether (ETH).
* It was proposed by Vitali Buterin in late 2013 and went live in 2015.
* Smart contracts run on the Ethereum virtual machine (EVM).
* Ether is the native cryptocurrency of the Ethereum platform. It’s used to pay for transaction fees and computational services on the network.

**Proof of History (POH):**

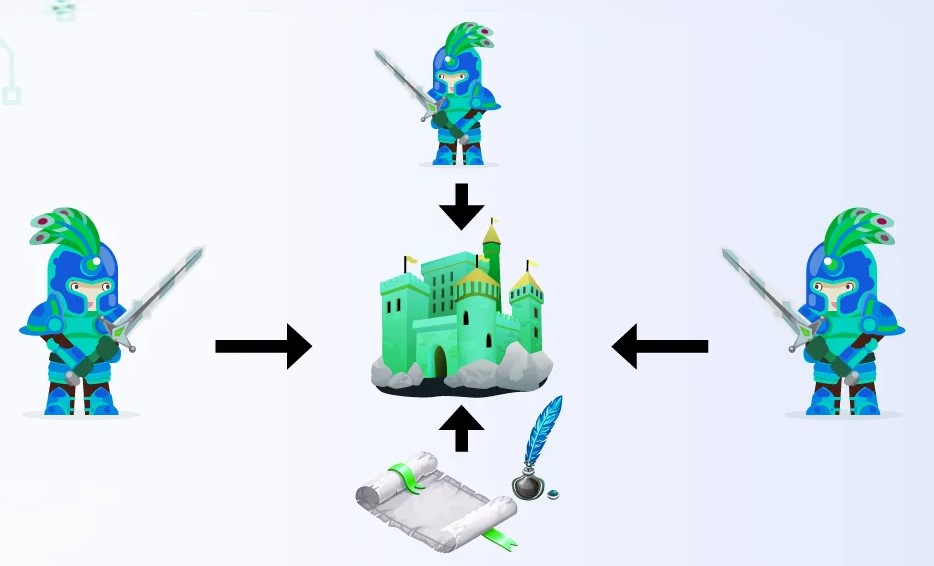
* PoH is a concept introduced by the Solana blockchain platform.
* It’s a mechanism designed to provide a verifiable and trustless source of time in a distributed system, such as a blockchain network.
* PoH is not a consensus mechanism itself but rather a component used alongside other consensus mechanisms, such as Proof of stake (Pos), to enhance the scalability and efficiency of a blockchain network.

**Byzantine Fault Tolerance (BFT):**

* Byzantine Fault Tolerance (BFT) is a property of distributed systems where the system continues to operate correctly and reach a consensus even if some of its components fail or behave maliciously. It ensures that the system can maintain reliability and consistency despite potential errors or attacks.

OR

* It is like having a group of friends working together on a project. Even if some friends make mistakes or give wrong information, the project can still be completed successfully because everyone double-checks and communicates effectively.
* Distributed system: Collection of independent computers that work together to achieve a common goal.
* Example:

****

* To attack the castle, they have to agree on the same point if one wants to attack but the other two don't want to attack, if one wants to send a message but the message gets corrupted or if two of them want to attack but only one says that.
* Therefore, it is difficult to reach an agreement on the same decision within a short period of time, and the time between discussions will pass so we need to focus on the majority.
* But the problem is how to deal with the nodes that provide false or misleading information to corrupt the block.
* Simple follow the majority.
* Faulty nodes should not be more than 1/3 of the entire population.

**Mempool or Memory Pool:**

* A backlog of pending and unconfirmed transactions in a blockchain.
* These unconfirmed transactions wait in the mempool to get validated and finalized in the upcoming block.

**Candidate Block:**

* It is a temporary block created by mining nodes using transactions from the memory pool.

**Zero - Knowledge Proofs:**

* A Zkp protocol is a method by which one party (the prover) can prove to another party (the verifier) that something is true, without revealing any information apart from the fact that this specific statement is true.
* Example:
  + One lady is driving and one cop stops her and says you need to prove your age. She shows her license to prove her age, but her license also contains other information like blood group and contact information.
  + We can say that in that case we have one black box and that girl just needs to add her age to that black box. The cop will ask for her age from that black box.
  + As the black box acts as a mediator between the girl and the cop, the black box acts as a zero-knowledge proof, and if the cop gets a yes, she'll be released.

**Verifiable Computation:**

* If we use too much information, then all the load gets to the blockchain so to remove the load we use a third-party
* blockchain does transactions through a third party with off-chain.
* Less execution for blockchain
* Reduce network congestion
* Improve transaction speed
* Usage:
  + Decentralized Identity
  + Medical records
  + Identity
  + Financial transaction
* Drawbacks:
  + Hardware cost
  + Proof of verification
  + Trust assumptions
  + Quantum computing threats

**VRF (Verifiable Random Function):**

* It is a cryptographic function that takes a series of inputs, computes them, and produces a pseudorandom output and proof of authenticity that can be verified by anyone.

**Blockchain Trilemma:**

* The blockchain trilemma refers to the challenge of balancing three critical properties in blockchain systems: decentralization, security, and scalability. Let's break it down:
  + Decentralization: This involves distributing control and decision-making power across a network of nodes rather than relying on a central authority. Decentralization enhances transparency, censorship resistance, and resilience against single points of failure.
  + Security: Security ensures the integrity and immutability of data stored on the blockchain. It involves cryptographic techniques, consensus algorithms, and network resilience to protect against unauthorized access, fraud, and tampering.
  + Scalability: Scalability refers to the ability of the blockchain network to handle a growing number of transactions or users without sacrificing performance. A scalable blockchain can process transactions quickly and efficiently while maintaining low fees and high throughput.
* Decentralization, scalability, and security cannot be achieved in a single blockchain network.
* The finance chain has scalability and security, but it doesn’t have scalability.
* Solution:
  + Lightning network (Off-chain solution)
  + Layer 2 scaling solutions
  + Consensus algorithm improvements
  + Sharding (shared into smaller, independent subsets called shards)
  + Optimization Techniques
  + Hybrid Approaches (combines multiple blockchains)

**Fungible Token:**

* Fungible assets are interchangeable and identical in nature.
* This means that one unit of fungible assets can be exchanged for another unit of the same asset without any loss of value or distinction.
* Example:
  + Traditional currencies like dollars, Rupees, Euros, etc.
  + Cryptocurrencies like Bitcoin or Ethereum (where each unit of the currency is identical and can be exchanged for another unit)
  + Commodities like gold or oil (Where each unit is the same as any other unit of the same type)
  + E.g. A, B, and C have 3 100 rupees notes so all the have same value.

**Non-Fungible Token:**

* Non-fungible assets are unique and distinct from each other.
* Each unit of a non-fungible asset has its own unique properties or characteristics that differentiate it from other units.
* Example:
  + Digital collectibles, such as crypto kitties or digital art where each item is unique and cannot be directly exchanged for another item without consideration of its individual properties.
* In simple terms:
  + Fungible assets are like identical Lego bricks, where you can exchange one brick for another, and they hold the same value.
  + Non-fungible assets are like unique paintings, where each painting is distinct and has its own value based on its individual characteristics.
  + In the context of blockchain technology, non-fungible tokens (NFTs) have gained popularity for representing ownership of unique digital assets, such as art, music, virtual real estate, and more.

**NFTs Usability:**

* Digital Art and Collectibles: NFTs gained widespread attention for their role in buying, selling, and owning digital art and collectibles. They enable artists to tokenize their work, proving ownership and authenticity in the digital realm.
* Gaming: NFTs can be used in gaming to represent unique in-game items, characters, or assets. Players can buy, sell, and trade these digital assets with ownership secured by blockchain technology. This can create new revenue streams for both game developers and players.
* Tokenizing Real-World Assets: NFTs can represent ownership of physical assets like real estate, luxury goods, or collectible items. By tokenizing these assets, ownership can be easily transferred, and fractional ownership can be facilitated.
* Intellectual Property and Royalties: NFTs can be used to manage intellectual property rights, allowing creators to track and monetize their digital content. Smart contracts embedded in NFTs can automate royalty payments to creators whenever their content is sold or used.
* Tickets and Events: NFTs can represent tickets for events, concerts, or conferences, ensuring authenticity, reducing fraud, and enabling easy transferability between individuals.
* Digital Identity: NFTs can serve as digital identity tokens, providing a secure and decentralized way to verify identity and personal information without relying on centralized authorities.
* Supply Chain and Provenance: NFTs can be used to track the provenance and ownership history of physical goods throughout the supply chain. This can help combat counterfeiting and ensure the authenticity of products.

**Fungible Vs Non-Fungible:**

|  |  |  |
| --- | --- | --- |
|  | **Fungible Tokens** | **Non-Fungible Tokens** |
| **Main Features** | Divisible | Indivisible |
| Non-unique | Unique |
| Interchangeable | Irreplaceable |
| **Real-world purposes** | Payment system store of value | Intellectual property |
|  | Academic title |
|  | Artwork |
|  | Music Composition |
|  | Gaming |
|  | Utility assets like stocks, shares |
| **Technology used** | Own blockchain | Built on another blockchain |
| **Example of tokens** | Bitcoin | ERC-721 |
| Litcoin |  |
| ERC-20 |  |
| **Content Stored** | Value | Data |

**Ethereum Improvement Proposal (EIP):**

* Ethereum Improvement Proposals (EIPs) are documents that outline standards, guidelines, or proposals for changes and enhancements to the Ethereum blockchain and its ecosystem. They provide a structured way for community members to propose, discuss, and implement changes.
* Idea Submission: The process begins with someone from the Ethereum community submitting an EIP. This could be an individual developer, a team, or even an organization. The EIP outlines the proposed change, improvement, or addition to the Ethereum protocol or ecosystem.
* Discussion: Once an EIP is submitted, it undergoes a period of discussion within the Ethereum community. This discussion typically takes place on forums, mailing lists, social media platforms, or specialized communication channels. Community members provide feedback, raise concerns, suggest improvements, and engage in debate about the proposal.
* Decision Making: After the discussion phase, the Ethereum community evaluates the proposal and decides whether to accept, reject, or modify it. This decision-making process often involves rough consensus, where there is general agreement among community members, although formal voting mechanisms may also be used in some cases.
* Implementation: If an EIP is accepted, the next step is to implement it. This involves writing the code changes necessary to incorporate the proposed improvement into Ethereum clients, smart contracts, developer tools, or other components of the ecosystem. Developers and teams responsible for Ethereum client software, such as Geth, Parity, or Besu, may work on implementing the EIP.
* Deployment: Once the implementation is complete, the updated software or protocol containing the EIP is deployed to the Ethereum network. This may involve releasing new versions of Ethereum client software, deploying updated smart contracts, or making changes to network parameters.
* Feedback and Improvement: After deployment, the Ethereum community continues to monitor the impact of the implemented EIP. Users, developers, and other stakeholders provide feedback based on their experiences using the new feature or change. This feedback may inform future iterations of the EIP or inspire new proposals for further improvements.

**Types of EIPs:**

* Core EIPs: These propose changes to the core protocol of the Ethereum blockchain, such as updates to consensus mechanisms, transaction processing, or network parameters.
* Networking EIPs: These focus on network communication protocols and peer-to-peer networking, aiming to enhance the efficiency, security, or scalability of Ethereum's network infrastructure.
* Interface EIPs: These specify standards and guidelines for client API specifications, user interface designs, or interactions with the Ethereum blockchain, making it easier for developers to create compatible software and tools.
* ERCs (Ethereum Request for Comments): ERCs define standards for smart contracts, token interfaces, or other Ethereum-based assets and functionalities, facilitating interoperability and compatibility across the Ethereum ecosystem.
* Meta EIPs: These address procedural issues, changes to the EIP process itself, or other meta-level concerns related to Ethereum governance, development, or community participation.

**ERC-20 [Fungible]:**

* Imagine you have a jar filled with identical marbles. Each marble is the same as the other.
* ERC-20 tokens are like those marbles.
  + They are digital tokens that are all the same.
  + E.g. if you have 10 tokens of a certain type, they’re all equal and can be exchanged interchangeably.
  + These tokens are often used for things like cryptocurrencies, loyalty points, or stablecoins, where each unit has the same value as any other unit.

**ERC-721 [non-fungible]:**

* Imagine you have a collection of unique trading cards; each card has different artwork & characteristic.
* ERC-721 tokens are like those trading cards. Each token is unique and distinct. They represent ownership of something specific, like digital art, collectibles, or virtual real estate.
* These tokens are used when each item has its value or identity, and they can’t be exchanged one-for-one with each other. Each token is special and can’t be replaced by another token of the same type.

**Solidity:**

* <https://docs.soliditylang.org/en/v0.8.25/index.html>