UNIT – 4 : ETHEREUM BLOCKCHAIN

Smart Contracts:

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

- The bitcoin network was the first to use some sort of smart contract by using them to transfer value from one person to another.
- The smart contract involved employs basic conditions like checking if the amount of value to transfer is actually available in the sender account.
- Later, the Ethereum platform emerged which was considered more powerful, precisely because the developers/programmers could make custom contracts in a Turing-complete language.
- It is to be noted that the contracts written in the case of the bitcoin network were written in a Turing-incomplete language, restricting the potential of smart contracts implementation in the bitcoin network.
- There are some common smart contract platforms like Ethereum, Solana, Polkadot, Hyperledger fabric, etc.

A smart contract is just a digital contract with the security coding of the blockchain.

- It has details and permissions written in code that require an exact sequence of events to take place to trigger the agreement of the terms mentioned in the smart contract.
- It can also include the time constraints that can introduce deadlines in the contract.
- Every smart contract has its address in the blockchain. The contract can be interacted with by using its address presuming the contract has been broadcasted on the network.

The idea behind smart contracts is pretty simple. They are executed on a basis of simple logic, IF-THEN for example:

- **IF** you send object A, **THEN** the sum (of money, in cryptocurrency) will be transferred to you.

- **IF** you transfer a certain amount of digital assets (cryptocurrency, for example, ether, bitcoin), **THEN** the A object will be transferred to you.
- **IF** I finish the work, **THEN** the digital assets mentioned in the contract will be transferred to me.

Note: The WHEN constraint can be added to include the time factor in the smart contracts. It can be seen that these smart contracts help set conditions that have to be fulfilled for the terms of the contract agreement to be executed. There is no limit on how much IF or THEN you can include in your intelligent contract.

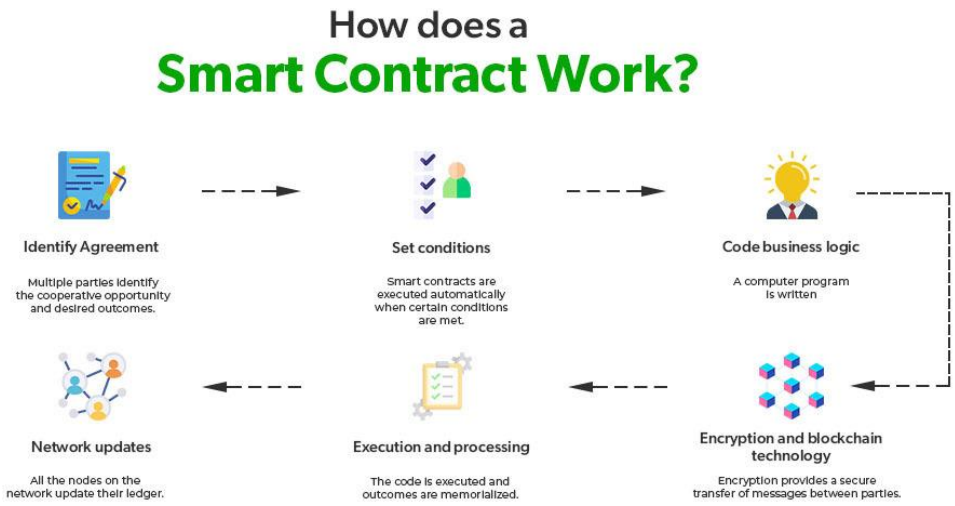
Features of Smart Contracts:

1. **Distributed:** Everyone on the network is guaranteed to have a copy of all the conditions of the smart contract and they cannot be changed by one of the parties. A smart contract is replicated and distributed by all the nodes connected to the network.
2. **Deterministic:** Smart contracts can only perform functions for which they are designed only when the required conditions are met. The final outcome will not vary, no matter who executes the smart contract.
3. **Immutable:** Once deployed smart contract cannot be changed, it can only be removed as long as the functionality is implemented previously.
4. **Autonomy:** There is no third party involved. The contract is made by you and shared between the parties. No intermediaries are involved which minimizes bullying and grants full authority to the dealing parties. Also, the smart contract is maintained and executed by all the nodes on the network, thus removing all the controlling power from any one party's hand.
5. **Customizable:** Smart contracts have the ability for modification or we can say customization before being launched to do what the user wants it to do.
6. **Transparent:** Smart contracts are always stored on a public distributed ledger called blockchain due to which the code is visible to everyone, whether or not they are participants in the smart contract.
7. **Trustless:** These are not required by third parties to verify the integrity of the process or to check whether the required conditions are met.
8. **Self-verifying:** These are self-verifying due to automated possibilities.
9. **Self-enforcing:** These are self-enforcing when the conditions and rules are met at all stages.

Capabilities of Smart Contracts:

1. **Accuracy:** Smart contracts are accurate to the limit a programmer has accurately coded them for execution.
2. **Automation:** Smart contracts can automate the tasks/ processes that are done manually.
3. **Speed:** Smart contracts use software code to automate tasks, thereby reducing the time it takes to maneuver through all the human interaction-related processes. Because everything is coded, the time taken to do all the work is the time taken for the code in the smart contract to execute.
4. **Backup:** Every node in the blockchain maintains the shared ledger, providing probably the best backup facility.
5. **Security:** Cryptography can make sure that the assets are safe and sound. Even if someone breaks the encryption, the hacker will have to modify all the blocks that come after the block which has been modified. Please note that this is a highly difficult and computation-intensive task and is practically impossible for a small or medium-sized organization to do.
6. **Savings:** Smart contracts save money as they eliminate the presence of intermediaries in the process. Also, the money spent on the paperwork is minimal to zero.
7. **Manages information:** Smart contract manages users' agreement, and stores information about an application like domain registration, membership records, etc.
8. **Multi-signature accounts:** Smart contracts support multi-signature accounts to distribute funds as soon as all the parties involved confirm the agreement.

Smart Contract Working:



- **Identify Agreement:** Multiple parties identify the cooperative opportunity and desired outcomes and agreements could include business processes, asset swaps, etc.
- **Set conditions:** Smart contracts could be initiated by parties themselves or when certain conditions are met like financial market indices, events like GPS locations, etc.
- **Code business logic:** A computer program is written that will be executed automatically when the conditional parameters are met.
- **Encryption and blockchain technology:** Encryption provides secure authentication and transfer of messages between parties relating to smart contracts.
- **Execution and processing:** In blockchain iteration, whenever consensus is reached between the parties regarding authentication and verification then the code is executed and the outcomes are memorialized for compliance and verification.
- **Network updates:** After smart contracts are executed, all the nodes on the network update their ledger to reflect the new state. Once the record is posted and verified on the blockchain network, it cannot be modified, it is in append mode only.

Applications of Smart Contracts:

1. **Real Estate:** Reduce money paid to the middleman and distribute between the parties actually involved. For example, a smart contract to transfer ownership of an apartment once a certain amount of resources have been transferred to the seller's account (or wallet).
2. **Vehicle ownership:** A smart contract can be deployed in a blockchain that keeps track of vehicle maintenance and ownership. The smart contract can, for example, enforce vehicle maintenance service every six months; failure of which will lead to suspension of driving license.
3. **Music Industry:** The music industry could record the ownership of music in a blockchain. A smart contract can be embedded in the blockchain and royalties can be credited to the owner's account when the song is used for commercial purposes. It can also work in resolving ownership disputes.
4. **Government elections:** Once the votes are logged in the blockchain, it would be very hard to decrypt the voter address and modify the vote leading to more confidence against the ill practices.
5. **Management:** The blockchain application in management can streamline and automate many decisions that are taken late or deferred. Every decision is transparent and available to any party who has the authority (an application on the private blockchain). For example, a smart contract can be deployed to trigger the supply of raw materials when 10 tonnes of plastic bags are produced.

6. **Healthcare:** Automating healthcare payment processes using smart contracts can prevent fraud. Every treatment is registered on the ledger and in the end, the smart contract can calculate the sum of all the transactions. The patient can't be discharged from the hospital until the bill has been paid and can be coded in the smart contract.

Example Use cases:

1. Smart contracts provide utility to other contracts. For example, consider a smart contract that transfers funds to party A after 10 days. After 10 days, the above-mentioned smart contract will execute another smart contract which checks if the required funds are available at the source account (let's say party B).
2. They facilitate the implementation of 'multi-signature' accounts, in which the assets are transferred only when a certain percentage of people agree to do so
3. Smart contracts can map legal obligations into an automated process.
4. If smart contracts are implemented correctly, can provide a greater degree of contractual security.

Advantages of Smart Contracts:

1. **Recordkeeping:** All contract transactions are stored in chronological order in the blockchain and can be accessed along with the complete audit trail. However, the parties involved can be secured cryptographically for full privacy.
2. **Autonomy:** There are direct dealings between parties. Smart contracts remove the need for intermediaries and allow for transparent, direct relationships with customers.
3. **Reduce fraud:** Fraudulent activity detection and reduction. Smart contracts are stored in the blockchain. Forcefully modifying the blockchain is very difficult as it's computation-intensive. Also, a violation of the smart contract can be detected by the nodes in the network and such a violation attempt is marked invalid and not stored in the blockchain.
4. **Fault-tolerance:** Since no single person or entity is in control of the digital assets, one-party domination and situation of one part backing out do not happen as the platform is decentralized and so even if one node detaches itself from the network, the contract remains intact.
5. **Enhanced trust:** Business agreements are automatically executed and enforced. Plus, these agreements are immutable and therefore unbreakable and undeniable.

6. **Cost-efficiency:** The application of smart contracts eliminates the need for intermediaries (brokers, lawyers, notaries, witnesses, etc.) leading to reduced costs. Also eliminates paperwork leading to paper saving and money-saving.

Challenges of Smart Contracts:

1. **No regulations:** A lack of international regulations focusing on blockchain technology (and related technology like smart contracts, mining, and use cases like cryptocurrency) makes these technologies difficult to oversee.
2. **Difficult to implement:** Smart contracts are also complicated to implement because it's still a relatively new concept and research is still going on to understand the smart contract and its implications fully.
3. **Immutable:** They are practically immutable. Whenever there is a change that has to be incorporated into the contract, a new contract has to be made and implemented in the blockchain.
4. **Alignment:** Smart contracts can speed the execution of the process that span multiple parties irrespective of the fact whether the smart contracts are in alignment with all the parties' intention and understanding.

Ethereum Structure:

What is Ethereum?

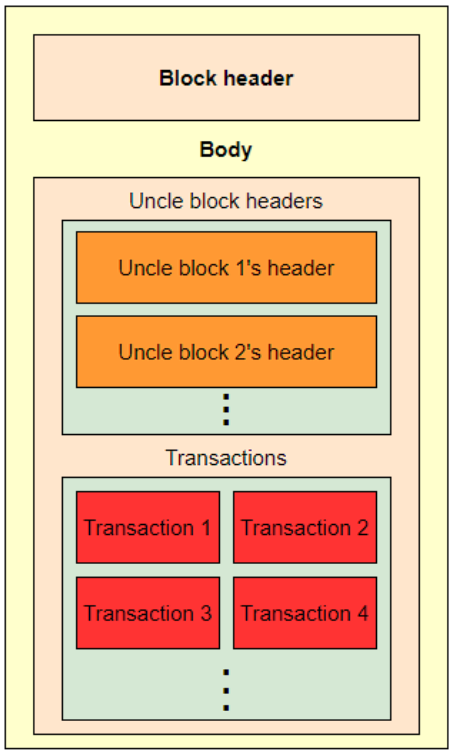
Ethereum is a Blockchain network that introduced a built-in Turing-complete programming language that can be used for creating various decentralized applications (also called Dapps). The Ethereum network is fueled by its own cryptocurrency called 'ether'.

- The Ethereum network is currently famous for allowing 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.
- Unlike bitcoin, Ethereum is a network that can be applied to various other sectors.
- Ethereum is often called Blockchain 2.0 since it proved the potential of blockchain technology beyond the financial sector.
- The consensus mechanism used in Ethereum is Proof of Stakes(PoS), which is more energy efficient when compared to that used in the Bitcoin network, that is, Proof of Work(PoW). PoS depends on the amount of stake a node holds.

Ethereum blocks store every transaction happening on the Ethereum network. All the blocks are chained together by adding the previous block's hash to the next block's header. This chaining makes the data in the previous blocks immutable because changing any information in the earlier blocks will change the hash of that block and cause unchaining of the blockchain.

Structure of Blocks:

Every block in the Ethereum blockchain follows a standard format, which is illustrated below:



An Ethereum block format

Every Ethereum block consists of two main parts:

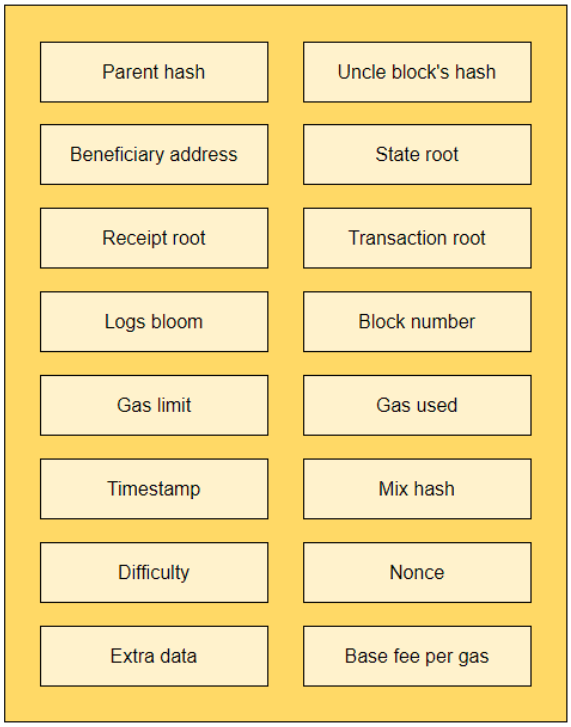
- Header
- Body

Header:

An Ethereum block header contains several fields that provide information about the block, miner, and current state of the network including:

- **Parent block's hash:** It is the Keccak 256-bit hash of its parent block's header (Last successfully mined block in the blockchain. This block becomes the parent of the block we are trying to mine). It is used to chain this block to the previous block making the blockchain more secure.
- **Uncle block's hash:** It is the Keccak 256-bit hash of the list of uncle block headers (Uncle blocks are stale blocks that do not make it to the actual blockchain. This happens when more than one miners mine a block at the same time.) present in the block's body.
- **Beneficiary address:** It is the 160-bit account address of the miner of this block to which all the mining fees from this block will be transferred.
- **State root:** It is the Keccak 256-bit hash of the root node of the state trie (It is a modified Merkle patricia trie that stores entire state of the system e.g. account balance, contract storage, contract code and account nonce of all accounts.) after all the transactions are executed and finalizations are applied.
- **Transaction root:** It is the Keccak 256-bit hash of the root node of the Merkle root trie populated by all the transactions present in that block's body.
- **Receipt root:** The Keccak 256-bit hash of the root of the Merkel root trie that is populated by the receipts of all the transactions in that block.
- **Logs bloom:** A Bloom filter consisting of indexable log entries from the receipt of each transaction from the list of transactions present in the body of the block.
- **Block number:** The length of the blockchain or the number of blocks in the blockchain. A positive whole number equals the number of ancestor blocks of this block where the genesis block is block 0.
- **Gas limit:** The current upper limit on the gas used in the block.
- **Gas used:** Total gas used in transactions in this block.
- **Difficulty:** It is the difficulty of the network to mine this block. Miners' have to mine a block whose header's hash is less than the network's difficulty at that time. Miners do this by changing the nonce value until they find a value with which the hash of the block is less than the network's difficulty.
- **Mix hash:** A unique identifier for the block. When combined with the nonce proves that a sufficient amount of work has been done by the miner (proof of work).
- **Timestamp:** The time in Unix format when the block was mined.
- **Base fee per gas:** The minimum fee required per gas for the transaction to be included in the block.
- **Extra data:** A byte array containing data relevant to this block. This must be less than 32 bytes.

The structure of the Ethereum block header is shown below:



An Ethereum block header

Body:

The body of an Ethereum block, also known as the "block payload" or "block data," is a collection of data that contains all the information necessary to execute the transactions included in the block. The main components of the block body are the list of transactions and the list of uncles (stale blocks). (both are described below).

- **Uncle block headers:** This is a list of uncle block headers (same format as a block header described above).

> **Note:** Every miner is trying to mine a block at the same time. In the end, only one block will be accepted and other blocks will become uncle blocks. In Ethereum uncle blocks also get paid.

- **Transactions:** This is a list of actual Ethereum transactions stored in the block.

Operations:

Operations in the Ethereum blockchain encompass a wide range of activities that contribute to the functioning and integrity of the network.

These operations include:

1. Transaction Processing:

- **Transaction Propagation**: Transactions are broadcasted across the network by nodes.
- **Transaction Validation**: Nodes verify the validity of transactions, checking for correct signatures, sufficient funds, and adherence to protocol rules.
- **Transaction Pool Management**: Valid transactions are collected into a pool (mempool) awaiting inclusion in a block.
- **Transaction Ordering**: Miners select transactions from the pool and order them into a block based on factors like gas price.

2. Block Creation and Mining:

- **Block Proposal**: Miners collect valid transactions and attempt to create a new block by solving a cryptographic puzzle (Proof of Work in Ethereum's current state).
- **Block Validation**: Other nodes validate the proposed block by checking its adherence to protocol rules and consensus mechanisms.
- **Block Addition**: If a proposed block is accepted by the network, it is added to the blockchain.

3. Gas Management:

- **Gas Calculation**: Each operation in Ethereum consumes a specific amount of gas, reflecting computational resources required.
- **Gas Limit Setting**: Users specify a gas limit for their transactions, which represents the maximum amount of gas they are willing to expend.
- **Gas Fees**: Users pay gas fees to miners for including their transactions in blocks. Fees are calculated as gas price * gas consumed.

4. Smart Contract Execution:

- **Smart Contract Deployment**: Developers deploy their smart contracts onto the Ethereum blockchain, defining their functionality and behavior.
- **Smart Contract Invocation**: Users interact with smart contracts by sending transactions that trigger their execution.
- **Smart Contract State Changes**: Smart contracts can modify the state of the Ethereum blockchain by updating account balances, storing data, or interacting with other contracts.

5. Consensus Mechanism:

- **Proof of Work (PoW)**: Ethereum currently uses PoW for consensus, where miners compete to solve mathematical puzzles to validate transactions and create new blocks.
- **Transition to Proof of Stake (PoS)**: Ethereum 2.0 aims to transition to a PoS consensus mechanism, where validators are chosen to create blocks based on the amount of cryptocurrency they hold and are willing to "stake" as collateral.

6. Decentralized Application (DApp) Support:

- **DApp Development**: Developers build decentralized applications on Ethereum using smart contracts for backend logic and the Ethereum blockchain for storage and execution.
- **User Interaction**: Users interact with DApps through their web browsers or specialized applications, sending transactions and interacting with smart contracts.
- **DApp Deployment and Hosting**: DApps are deployed on the Ethereum blockchain and can be accessed by users worldwide without the need for central servers.

These operations collectively form the backbone of Ethereum's functionality, enabling decentralized finance, tokenization, gaming, and various other applications on the blockchain.

Consensus Model:

The Ethereum blockchain currently uses a proof-of-work (PoW) consensus model called Ethash, which is similar to Bitcoin's consensus mechanism. However, Ethereum is in the process of transitioning to a proof-of-stake (PoS) consensus model called Ethereum 2.0 or Serenity.

Here's an overview of the consensus models in Ethereum:

1. Proof-of-Work (PoW) - Ethash:

- Ethash is the current PoW consensus algorithm used by Ethereum.
- Miners compete to solve complex cryptographic puzzles by performing computationally intensive calculations.
- The first miner to solve the puzzle gets to create a new block and receive a reward in Ether (ETH).
- Ethash is designed to be memory-hard, making it more resistant to specialized mining hardware (ASICs) and promoting a more decentralized mining ecosystem.

2. Proof-of-Stake (PoS) - Ethereum 2.0:

- Ethereum is transitioning to a PoS consensus model called Ethereum 2.0 or Serenity.
- Instead of mining, validators will stake their ETH as collateral to participate in the consensus process.
- Validators are chosen randomly to propose and attest to new blocks based on their stake (number of ETH held).
- PoS aims to be more energy-efficient, scalable, and secure compared to PoW.
- Ethereum 2.0 will introduce sharding, which partitions the network into multiple shard chains to improve scalability and throughput.

The transition from PoW to PoS is happening gradually through a multi-phase process called the Ethereum Serenity upgrade. The phases include:

1. **The Beacon Chain**: A separate PoS blockchain initially launched in December 2020, serving as the coordination hub for validators and the future Ethereum 2.0 system.
2. **The Merge**: This phase will merge the existing PoW Ethereum mainnet with the Beacon Chain, transitioning Ethereum to a full PoS consensus.
3. **Shard Chains**: After the merge, shard chains will be introduced, allowing for parallel transaction processing and increasing the scalability of the Ethereum network.

The PoS consensus model in Ethereum 2.0 aims to address some of the scalability and energy efficiency limitations of the current PoW system while maintaining the core principles of decentralization and security. However, the transition is a complex process and is being approached cautiously to ensure a smooth migration and maintain the integrity of the Ethereum network.

Incentive Model:

The Ethereum blockchain utilizes an incentive model to encourage participation and maintain the security and integrity of the network. The incentive model is designed to reward participants, such as miners (in the current proof-of-work system) and validators (in the upcoming proof-of-stake system), for their contributions to the network.

1. **Proof-of-Work (PoW) Incentive Model**:

- **Block Rewards**: Miners who successfully mine a new block are rewarded with newly minted Ether (ETH) and the transaction fees (gas fees) included in the block.
- **Mining Rewards**: The current block reward for mining a block is approximately 2 ETH, which is earned by the miner who solves the cryptographic puzzle first.
- **Transaction Fees**: Miners also receive the transaction fees (gas fees) paid by users for including their transactions in the block.

2. **Proof-of-Stake (PoS) Incentive Model (Ethereum 2.0)**:

- **Staking Rewards**: Validators who stake their ETH and participate in the consensus process by attesting to and proposing new blocks will earn rewards in the form of newly minted ETH.
- **Transaction Fees**: Similar to the PoW model, validators will also receive a portion of the transaction fees (gas fees) for processing transactions.
- **Penalties and Slashing**: To maintain the integrity of the network, validators can be penalized or have a portion of their staked ETH slashed if they engage in malicious behavior, such as proposing invalid blocks or going offline for an extended period.

In both the PoW and PoS models, the incentive structure is designed to encourage participants to act in the best interest of the network. By contributing their computational resources (in PoW) or staked ETH (in PoS), participants are rewarded with newly minted ETH and transaction fees.

The transition to Ethereum 2.0 and the PoS consensus model aims to improve the incentive structure by making it more energy-efficient and reducing the need for specialized mining hardware. Additionally, the introduction of penalties and slashing mechanisms in PoS is intended to discourage malicious behavior and ensure the network's security and reliability.

It's important to note that the incentive model is subject to ongoing development and may be adjusted or refined as Ethereum continues to evolve and implement new upgrades and features.