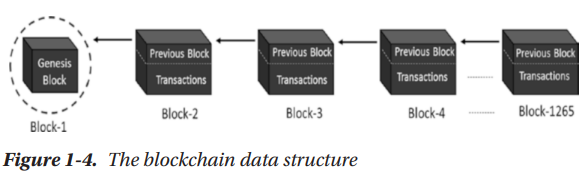
### **UNIT 1**

**Ref:** Chp 1,2,3 “Beginning Blockchain” A Beginner’s Guide to Building Blockchain Solutions

**1. What is Blockchain?**

* Blockchain is a peer-to-peer system of transacting values without any third parties involve in between.
* It is a shared, decentralized, and open ledger of transactions. This ledger database is replicated across a large number of nodes.
* This ledger database is an append-only database and cannot be changed or altered. It means that every entry is a permanent entry. Any new entry on it gets reflected on all copies of the databases hosted on different nodes.
* It is another layer on top of the Internet and can coexist with other Internet technologies.
* A typical blockchain may look as shown as below



* The first block of any blockchain is called as genesis block. Each block has two main parts: the "header" and the "body."
* The header links to the previous block by storing its hash, which prevents anyone from changing past transactions.
* The body contains a list of validated transactions, including amounts, addresses, and other details. This structure allows you to trace all previous blocks in the blockchain from the latest one.
* The transaction data in the blocks is immutable. All transactions are fully irreversible. Any change would result in a new transaction, which would get validated by all contributing nodes. Every node has its own copy of blockchain.

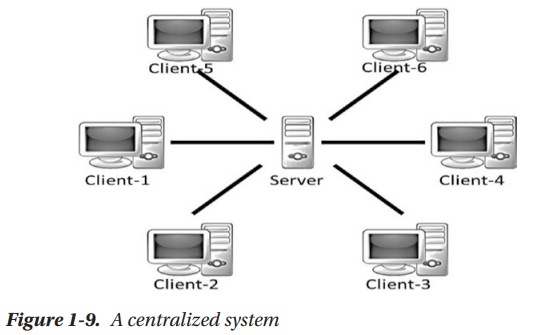
**2. Centralized vs Decentralized systems**

**Centralized Systems**

A centralized system has a centralized control with all administrative authority to a single entity. Such systems are easy to design, maintain, impose trust, and govern. They have a central point of failure, so they are less stable as well.

Limitation of centralized system

* They are more vulnerable to attack and hence less secure.
* Centralization of power can lead to unethical operations.
* Scalability is difficult most of the time.

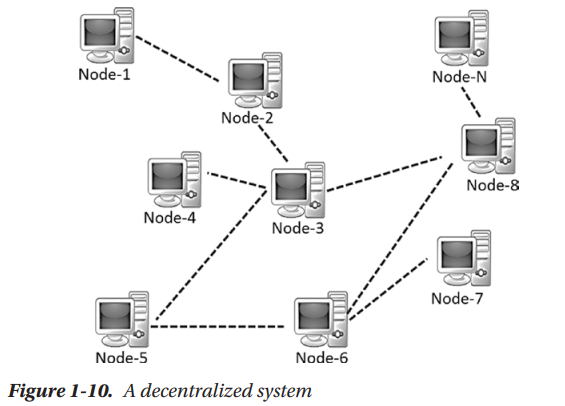


**Decentralized Systems**

A decentralized system does not have a centralized control and every node has equal authority. Such systems are difficult to design, maintain, govern, or impose trust. However, they do not suffer from the limitations of conventional centralized systems.

Decentralized systems offer the following advantages:

* They do not have a central point of failure, so more stable and fault tolerant
* Attack resistant, as no central point to easily attack and hence more secured
* Symmetric system with equal authority to all, so less scope of unethical operations and usually democratic in nature



**3. Limitations of Centralized Systems**

Limitations of a centralized system:

* Trust issues
* Security issue
* Privacy issue: data sale privacy is being undermined
* Cost and time factor for transactions

**4. Advantages of Decentralized systems over centralized system**

Advantages of decentralized systems over centralized systems

* Elimination of intermediaries
* Easier and genuine verification of transactions
* Increased security with lower cost
* Greater transparency
* Decentralized and immutable

**5. What are 5 layers of Blockchain?**

**6. Write a short note on**

(i) Application Layer

(ii) Execution Layer

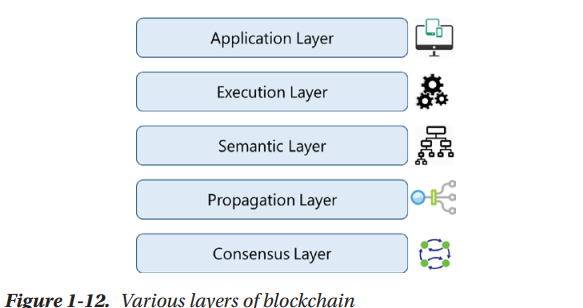
(iii) Semantic Layer

(iv) Propagation Layer

(v) Consensus Layer

**Q5 & Q6**

* Blockchain technology doesn’t have global standards yet to clearly separate its components into different layers. However, to better understand and compare the many types of blockchains and cryptocurrencies, we can think of blockchain as having different layers.
* While the exact structure is still being developed, this approach helps explain how blockchain works and how different systems are built.



**Layers of Blockchain (AES PC)**

1. **Application Layer**

* This layer is where **blockchain applications are built**, utilizing the features of blockchain like immutability, transparency, and resilience.
* It involves traditional software development tools like APIs, scripting, and client-server programming, although ideal blockchain applications are decentralized, similar to Bitcoin.

1. **Execution Layer**

* This layer is **responsible for executing instructions**, often **through smart contracts**, on all nodes in the blockchain network.
* It ensures **deterministic execution**, meaning the **same inputs always produce the same output** across all nodes.

1. **Semantic Layer**

* The Semantic Layer **validates the transactions** and ensures the integrity of the blockchain’s state.
* This layer also **defines data models, smart contract rules, and how blocks are linked through hash functions,** ensuring the blockchain’s consistency.

1. **Propagation Layer**

* This peer-to-peer layer allows nodes to **discover, communicate, and synchronize with other nodes**, ensuring the latest transactions and blocks are shared across the network.
* It **manages the broadcasting of transactions and blocks**, which helps maintain network stability, although latency may vary depending on node capacity and network conditions.

1. **Consensus Layer**

* The consensus process is essential for the blockchain’s integrity, as it **involves selecting nodes to propose blocks and validating the correctness of these blocks** across the network.
* It uses **mechanisms like Proof of Work (PoW) or Proof of Stake (PoS)** to achieve consensus and secure the network.

1. **Application Layer:** Blockchain apps, decentralized, off-chain tasks.
2. **Execution Layer:** Instruction execution, deterministic, smart contracts.
3. **Semantic Layer:** Transaction validation, integrity, data models.
4. **Propagation Layer:** Node communication, synchronization, broadcasting.
5. **Consensus Layer:** Node agreement, consensus mechanisms, security.

7. Why is Blockchain Important? Give some practical use cases of Blockchain

**8. Explain: (i) Stream Ciphers (ii) Block Cipher**

**Stream Ciphers**

* Stream ciphers convert one symbol of plaintext into one symbol of ciphertext, encrypting data one bit or byte at a time.
* To encrypt every bit of plaintext, a different key is generated and used, creating an infinite stream of pseudorandom bits as the key.
* The keystream is generated serially from a random seed value using digital shift registers. Stream ciphers are simple and faster in execution, allowing for offline keystream generation and quick decryption, but they require synchronization in most cases.
* The security of a stream cipher depends on the pseudorandom number generator used to create the keystream, and vulnerabilities in the generator have led to the deprecation of some stream ciphers.
* One notable example is RC4, which was widely used but is now considered insecure.
* Stream ciphers also suffer from low diffusion, where information from one bit of plaintext is directly reflected in its corresponding ciphertext bit, unlike block ciphers that offer better diffusion.

**Block Cipher**

* Block cipher is based on the idea of partitioning the plaintext into fixed-length blocks and encrypting each block separately using the same key.
* It is a deterministic algorithm, meaning that when the same plaintext block is encrypted with the same key, the result will always be the same.
* Block sizes are typically 64 bits, 128 bits, or 256 bits, and the ciphertext blocks are the same size as the plaintext blocks. The key used to encrypt each block is chosen randomly, but the key space is limited by the size of the key, so the notion of a “perfect cipher” does not apply.
* To encrypt or decrypt data larger than a block, a block cipher operates in different "modes of operation."
* These modes, such as Electronic Codebook (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB), Output Feedback (OFB), and Counter (CTR), help mix plaintext blocks with ciphertext to prevent repeated patterns and improve security.
* The primary purpose of these modes is to avoid the deterministic nature of block ciphers, where the same plaintext block encrypted with the same key would produce identical ciphertext, leaking valuable information.
* Block ciphers are generally slower than stream ciphers and are more prone to error propagation, where an error in one bit could corrupt the entire block.
* Block ciphers have the advantage of high diffusion, meaning that each input bit is spread out across many ciphertext bits, providing stronger security.
* Examples of block ciphers include DES, 3DES, and AES.

**9. Explain:**

**(i) Symmetric Key Cryptography (ii) Asymmetric/Public Key Cryptography**

**Symmetric Key Cryptography**

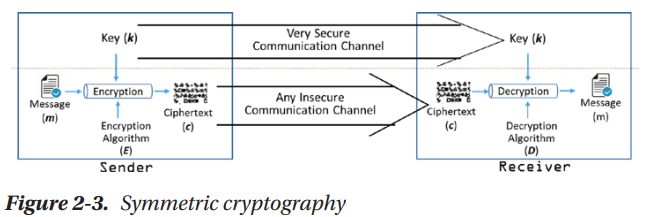
* If the same key is used for both encryption and decryption, it is called symmetric key cryptography.
* This means that both **sender and receiver have to agree on a key (k) called “shared secret” before they exchange the ciphertext.** So, the process is as follows:

**Sender**

* Encrypt the plaintext message m using encryption algorithm E and key k to prepare the ciphertext c
* c = E(k, m)
* Send the ciphertext c to Bob

**Receiver**

* Decrypt the ciphertext c using decryption algorithm D and the same key k to get the plaintext m
* m = D( k, c )



* Symmetric key cryptography exists in **two variants: stream ciphers and block ciphers.**
* Symmetric key cryptography is used widely; the most common **uses are secure file transfer protocols.**
* Symmetric cryptosystems are usually **faster** and more **useful when the data size is huge.**

(ii) Asymmetric/Public Key Cryptography

10. Write a short note on Digital Signature

* Same use-case/cryptographic method (ECDSA) as BTC
* Signer uses private key to generate a signed message
* Signed message can be verified using the signer’s public key
* Hashes are signed in Ethereum, not the data itself

11. What is the Data Encryption Standard (DES)? Explain One round of DES Cryptography

12. What is a Cryptographic Key and Its properties? Define Public and Private Key

13. Explain RSA with Example

**14. What is Hash Function? Explain with Example**

A hash function is a process that takes an input (or "message") and turns it into a fixed-size string of characters, which is typically a sequence of numbers and letters. This output is called a "hash value" or "digest."

* BTC uses SHA-256
* Ethereum uses Keccak-256
  + Similar to SHA-3 (variant)
  + Won contest for security in 2007
  + Used for all hashing in Ethereum
  + Derived differently than standard block-cipher based hashes or previous SHA functions

15. What are the various SHA-Secure Hash Algorithms

16. What are the applications and advantages of Hash Function

17. What are the various applications of blockchain

18. Explain Byzantine Generals’ Problem

19. Write a short note on Merkle Trees

20. What are the various properties of Blockchain Solutions

21. Write a short note on Blockchain Transactions

22. Explain the Distributed Consensus Mechanisms

23. Write a short note on

(i) Proof of Work

(ii) Proof of Stake

(iii) Proof of Burn

24. What do you mean by Scaling Blockchain? Explain any one of the Scaling Techniques

(i) Off-Chain Computation

(ii) Sharding

25. Define Bitcoin? How is it different from standard government-issued currency?

26. What is Bitcoin? How does it Work?

27. Explain the bitcoin blockchain block structure

28. How is Merkle tree used for blockchain explain with example

29. Write a short note on the genesis block

30. Explain the bitcoin network? How does a new node join the network?

31. Write a short note bitcoin transaction

32. What is Bitcoin Mining?

33. Benefits and limitations of blockchain

34.Challenges and limitations of blockchain technology

### **UNIT 2**

Refer: PPT and <https://ethereum.org/developers/docs/intro-to-ethereum>

<https://medium.com/@preethikasireddy/how-does-ethereum-work-anyway-22d1df506369>

1. Explain how Ethereum Blockchain Works.
2. Explain two Types of accounts in Ethereum Blockchain
3. Explain about three different Merkle trie structures in Ethereum
4. Write a short note on gas and Payment.
5. Explain transaction and messages in Ethereum
6. Describe fields of block header in Ethereum blockchain
7. Write a short note on EVM( Ethereum Virtual Machine)
8. Write short note on Ommer Blocks
9. Explain Ethereum Smart Contract
10. **Explain the consensus mechanism**

(PoW) Proof of Work

(PoS) Proof of Stake

(POB) Proof of Burn

A consensus mechanism is a way for all participants in a blockchain network to agree on the validity of transactions and the state of the system. It ensures the network is secure, decentralized, and everyone follows the same rules without needing a central authority.

**1. Proof of Work (PoW)**

* Proof of Work is a consensus mechanism used to validate transactions and secure the network. It requires participants (miners) to solve complex cryptographic puzzles before they can add a new block to the blockchain.
* **How it works**:
  + Miners compete to solve the puzzle by trying different nonce values, which are random numbers that are included in the hashing process.
  + The first miner to solve the puzzle gets to add the block to the blockchain and is rewarded with cryptocurrency (e.g., Bitcoin).
  + This process consumes a lot of computational power and electricity, as miners need high-performance hardware to solve the puzzles efficiently.
* **Pros**:
  + Highly secure, since altering the blockchain requires redoing the work for all subsequent blocks.
  + Decentralized, as anyone can participate in mining if they have the necessary resources.
* **Cons**:
  + Energy-intensive and environmentally costly due to the hardware requirements.
  + Mining can become centralized in areas with cheaper electricity, potentially leading to centralization of control.

**2. Proof of Stake (PoS)**

* Proof of Stake is a more energy-efficient alternative to Proof of Work. In PoS, validators (instead of miners) are chosen to create new blocks based on how much cryptocurrency they "stake" or lock up as collateral.
* **How it works**:
  + Validators are selected to propose a new block based on the amount of cryptocurrency they have staked and sometimes their age or other factors.
  + Once a block is proposed, other validators confirm the block’s legitimacy, and the validator who proposed the block receives a reward (usually transaction fees or new cryptocurrency issuance).
  + If a validator acts maliciously (e.g., tries to create fraudulent blocks), they lose a portion of their staked funds as a penalty.
* **Pros**:
  + Much more energy-efficient than PoW, since it doesn’t require intensive computational work.
  + Can be more secure against attacks, as attacking the network would require a large portion of the staked cryptocurrency.
* **Cons**:
  + Wealth concentration: validators with more coins staked have a higher chance of being chosen, which could lead to wealth centralization and inequality.
  + Newer and still evolving, meaning it’s more experimental in some blockchain ecosystems.

**3. Proof of Burn (PoB)**

* Proof of Burn is a less common consensus mechanism that involves participants burning (i.e., permanently destroying) a portion of their cryptocurrency to prove their commitment to the network.
* **How it works**:
  + To participate in the creation of new blocks, participants send their cryptocurrency to an address where it is "burned" (made inaccessible to anyone).
  + The more cryptocurrency a participant burns, the higher their chances of being chosen to create the next block.
  + Burning cryptocurrency demonstrates that the participant has invested value into the system, and this "commitment" helps secure the network.
* **Pros**:
  + Energy-efficient compared to PoW because it doesn’t require intense computation.
  + Reduces supply of the cryptocurrency, which can potentially increase scarcity and value.
* **Cons**:
  + Cryptocurrency burned is permanently lost, which might not seem like the best long-term strategy for participants.
  + Still a relatively untested mechanism compared to PoW and PoS.

**Solidity Programming**

Ref: [Solidity - Overview (tutorialspoint.com)](https://www.tutorialspoint.com/solidity/solidity_overview.htm)

Solidity is a contract-oriented programming language used for writing smart contracts on the Ethereum blockchain. It is statically typed and supports inheritance, libraries, and complex user-defined types.

1. Solidity - Basic Syntax
2. Solidity - First Application
3. Solidity - Comments
4. Solidity - Types
5. Solidity - Variables
6. Solidity - Variable Scope
7. Solidity - Operators
8. Solidity - Loops
9. Solidity - Decision Making
10. Solidity - Strings
11. Solidity - Arrays
12. Solidity - Enums
13. Solidity – Structs
14. Visibility Quantifiers in Solidity
15. Solidity - Mappings
16. Solidity - Conversions
17. Solidity - Ether Units
18. Solidity - Special Variables
19. Solidity Functions
20. Solidity - Functions
21. Solidity - Function Modifiers
22. Solidity - View Functions
23. Solidity - Pure Functions
24. Solidity - Fallback Function
25. Function Overloading
26. Mathematical Functions
27. Cryptographic Functions
28. Solidity - Contracts
29. Solidity - Inheritance
30. Solidity - Constructors
31. Solidity - Abstract Contracts
32. Solidity – Interfaces
33. Difference between Memory and Storage and Stack in Solidity

Ref : [Memory vs storage in Solidity tutorials with examples (w3schools.io)](https://www.w3schools.io/blockchain/solidity-memory-vs-storage/)

UNIT III

Smart Contracts and Token

1. Define Token and types of TOKEN standard in Ethereum

Ref: <https://crypto.com/university/what-are-token-standards>

Explain ERC-20 TOKEN standard

Explain ERC-721 TOKEN standard

Explain ERC-1155 TOKEN standard

Explain ERC-4626 TOKEN standard

Ref: <https://ethereum.org/developers/docs/standards/tokens>

1. What are digital Collectibles
2. What Are the Benefits of Owning Digital Collectibles?

Ref : <https://ucollex.io/blog/what-are-digital-collectibles>

<https://medium.com/crypto-simplified/a-simple-explanation-of-crypto-collectibles-8674c4527bd1>

1. What are Fungible and Non-Fungible Tokens
2. What are the differences between Fungible and Non-Fungible Tokens
3. What is a crypto token?

7.What are the Features of a crypto token

Ref: <https://www.vestinda.com/blog/creating-a-crypto-token-a-step-by-step-guide>

8. How to build and deploy your own ERC-20 token

Ref: <https://www.quicknode.com/guides/ethereum-development/smart-contracts/how-to-create-and-deploy-an-erc20-token>

9.Explain the different tokens available

(payment tokens, stablecoins, utility tokens, security tokens, and non-fungible tokens.)

Blockchain Application Development

Ref Book: “Beginning Blockchain” Chapter 5

1. Explain the steps followed for Interacting with the Bitcoin Blockchain.

(Setup and Initialize the bitcoinjs Library in a node.js Application, Create Keypairs for the Sender and Receiver, Get Test Bitcoins in the Sender’s Wallet ,Get the Sender’s Unspent Outputs ,Prepare Bitcoin Transaction, Sign Transaction Inputs, Create Transaction Hex, Broadcast Transaction to the Network)

1. Explain the steps to Interact Programmatically with Ethereum Blockchain.

(Set Up Library and connection, Set Up Ethereum Accounts, Get Test Ether in Sender’s Account, Prepare Ethereum Transaction, Sign Transaction,

Send Transaction to the Ethereum Network)

1. Explain how to create a smart contract.

(Program the Smart Contract, Compile Contract and Get Details, Deploy Contract to Ethereum Network)

1. Explain Public Nodes vs. Self-Hosted Nodes
2. Differentiate between public, private and Consortium blockchain.
3. Explain the merits and demerits of public, private and Consortium blockchain.

Ref: <https://www.blockchain-council.org/blockchain/types-of-blockchains-explained-public-vs-private-vs-consortium/>

**UNIT IV**

DAPP Deployment

Ref Book : Building Ethereum Dapps -Roberto Infante Manning Publications-

Chapter 1 & Chapter 2

1. Define DAPP. What are the benefits of DApp development
2. What are the drawbacks for Dapp deployment?
3. Difference between Dapps and conventional centralized applications.
4. Explain the structural Anatomy of a Dapp (or) Explain the structural Anatomy of a Voting Dapp.
5. Explain the step by step full cycle of voting transactions.
6. Explain the components of Ethereum node.
7. Explain how an Ethereum node components - JSON-RPC interface, an EVM, a memory pool play a important role in lifecycle of transaction.
8. Explain the Development view of Deploying the voting smart contract.
9. Explain The Architecture of a Web 3.0 application

Ref : [The Architecture of a Web 3.0 application (preethikasireddy.com)](https://www.preethikasireddy.com/post/the-architecture-of-a-web-3-0-application)

1. Explain JSON RPC in detail

Ref: <https://www.wallarm.com/what/what-is-json-rpc>

Uses Cases of Blockchain

Ref Textbook: Chapter 10 : Introducing Ethereum and Solidity

Foundations of Cryptocurrency and Blockchain Programming for Beginners — Chris Dannen

1. Explain the use of Blockchain in The Internet of Ethereum Things.
2. Explain the use of Blockchain in Retail and E-Commerce.
3. Explain the use of Blockchain in Community and Government Financing.
4. Explain the use of Blockchain in Human and Organizational Behavior.
5. Explain the use of Blockchain in Financial and Insurance Applications.
6. Explain the use of Blockchain in Inventory and Accounting Systems.
7. Explain the use of Blockchain in Software Development.
8. Explain the use of Blockchain in Gaming, Gambling, and Investing.
9. Explain the use of Blockchain in Supply chain Management.
10. Explain the role of blockchain in healthcare.