Blockchain Technology

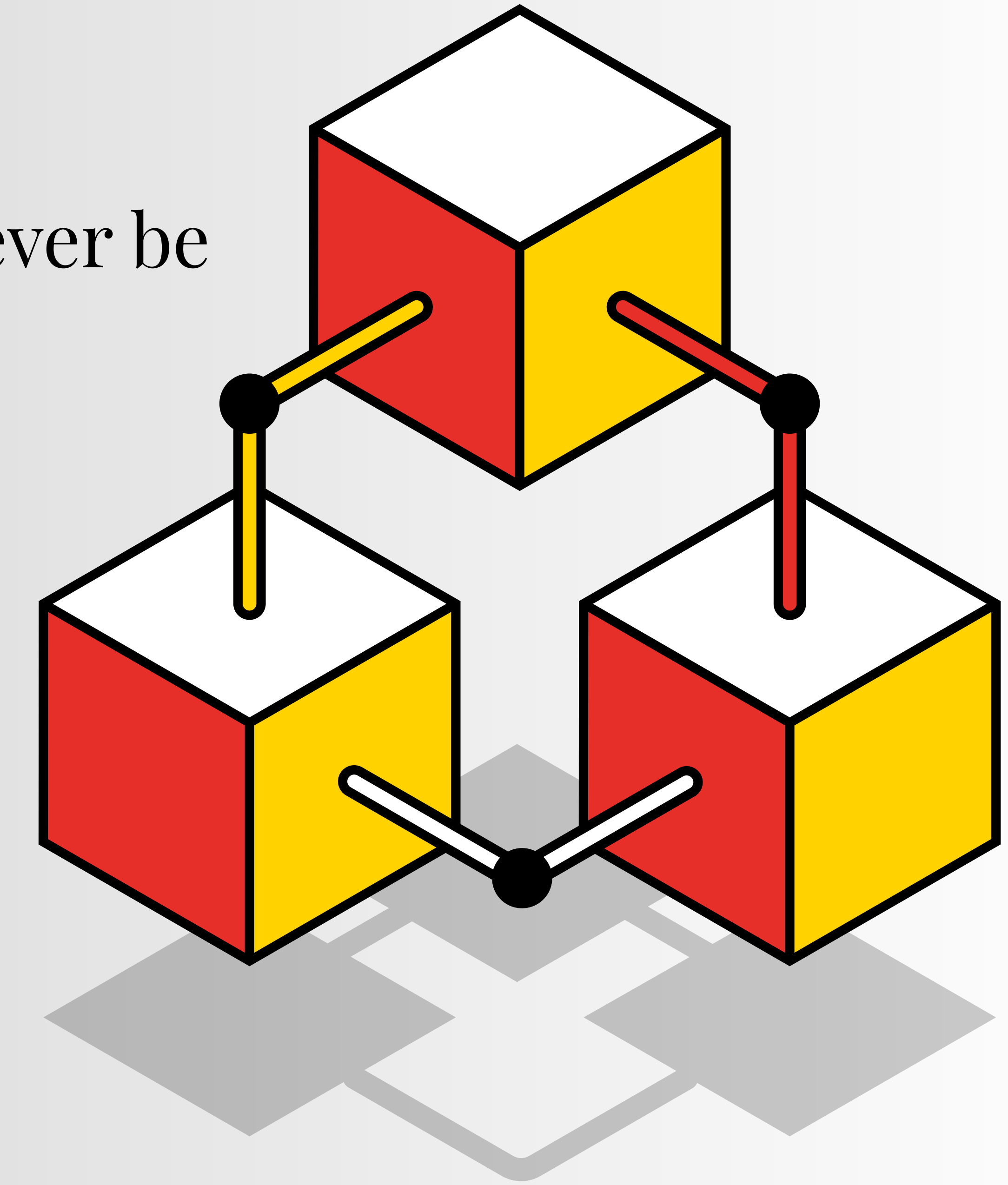
Feature Engineering

UNIT 2

Contents

Introduction to Blockchain

- **Ledger**: File that is constantly growing.
- **Permanent**: The transaction is stored permanently in the ledger.
- **Secure**: It uses advanced cryptography to lock the information inside the blockchain.
- **Chronological**: Every transaction happens after the previous one.
- **Immutable:** Once transaction stored onto the blockchain , it can never be changed.



History Of Blockchain

1991 – Beginning

- Stuart Haber & W. Scott Stornetta created digital timestamping to prevent document tampering.
- Example: Proving a research paper’s date.

2008–2013 – Blockchain 1.0 (Digital Money)

- Bitcoin by Satoshi Nakamoto enabled peer-to-peer money transfer without banks.
- Example: Sending money abroad without PayPal or banks.

2014–2017 – Blockchain 2.0 (Smart Contracts)

- Ethereum introduced smart contracts (self-executing rules).
- Example: Auto ticket refund if train is late.

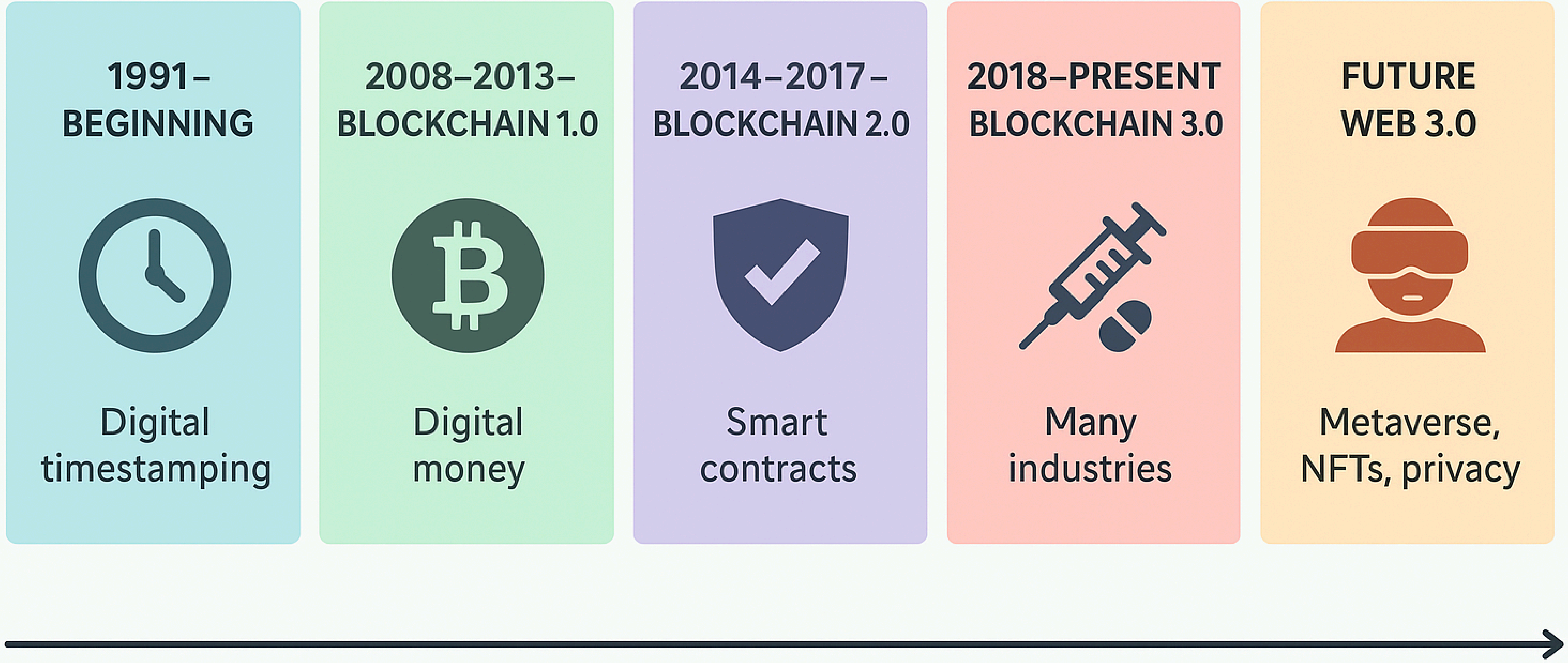
2018–Present – Blockchain 3.0 (Industries)

- Expanded to healthcare, supply chain, banking, voting.
- Example: Tracking genuine medicines from factory to pharmacy.

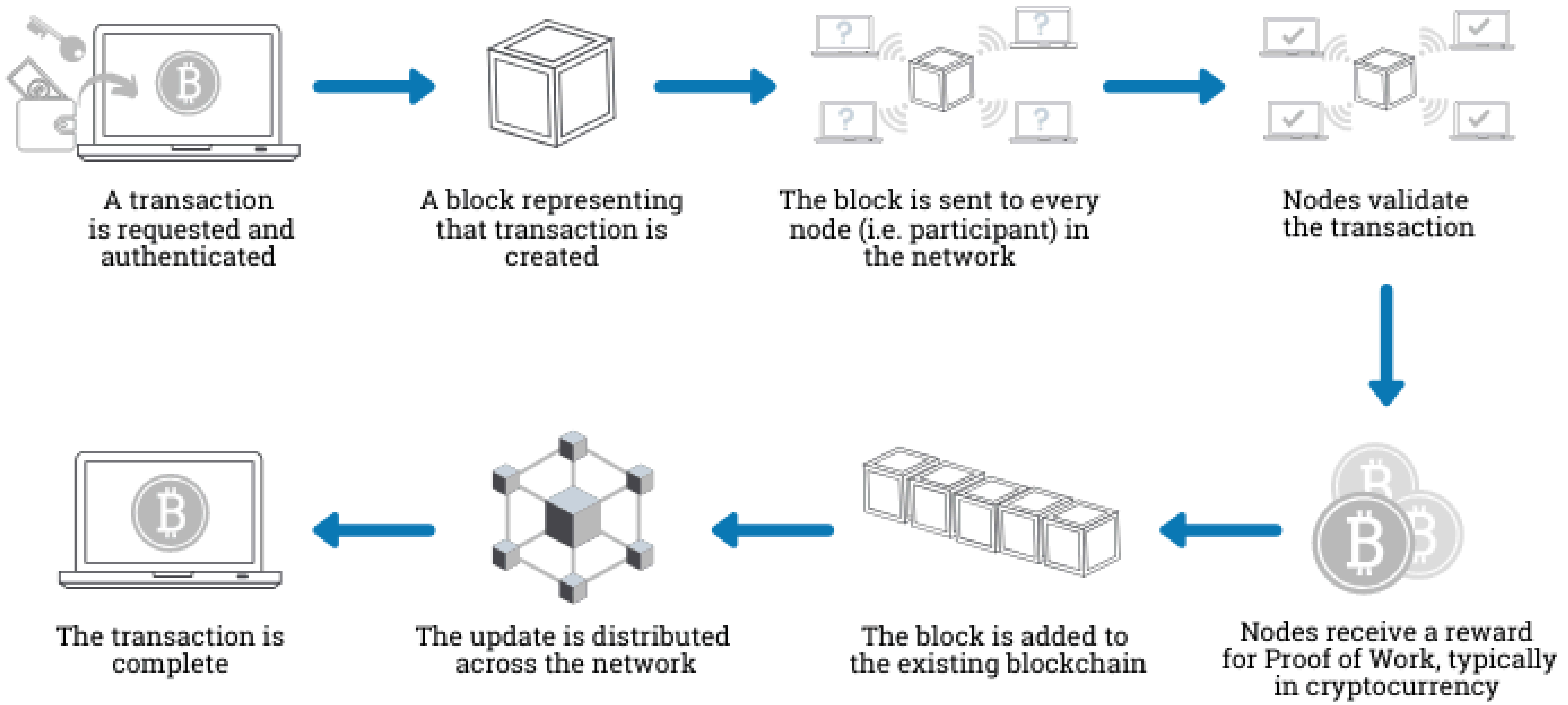
Future – Web 3.0 & Beyond

- Focus on Metaverse, DAOs, NFTs, privacy, decentralized internet.
- Example: Owning & selling digital art (NFTs) directly.

BLOCKCHAIN ADOPTION TIMELINE



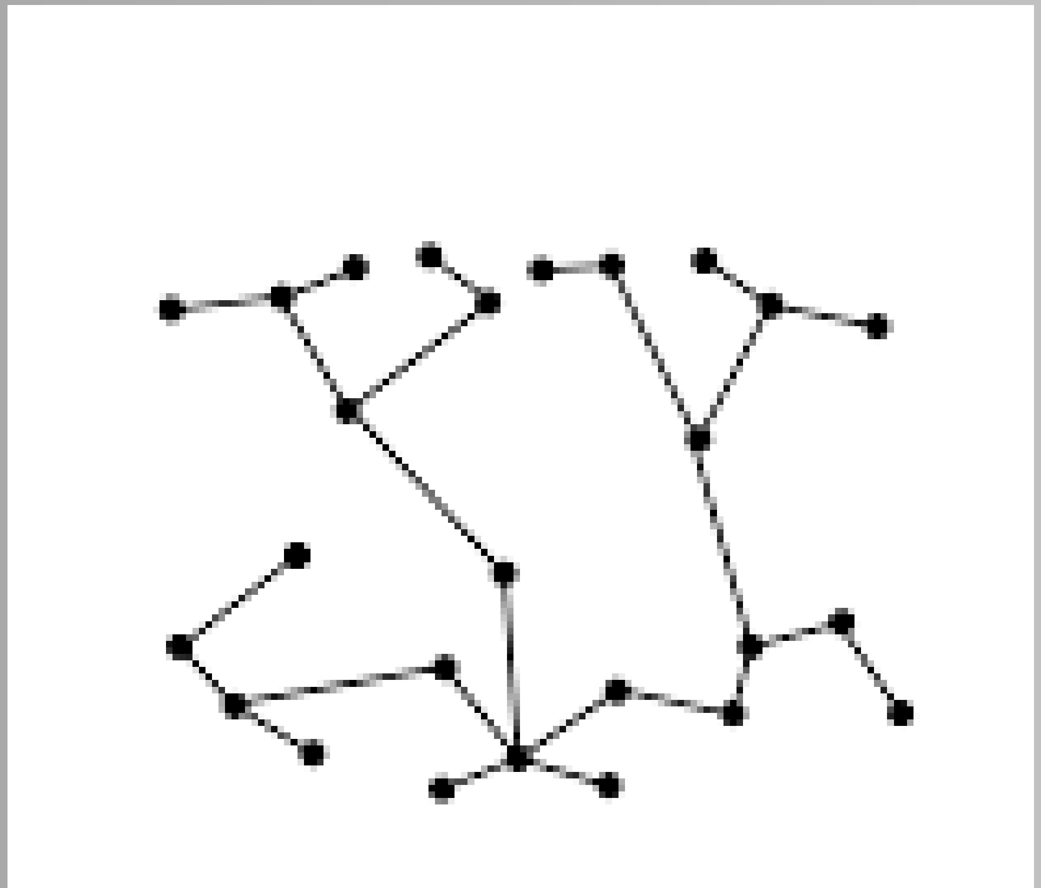
How does blockchain works?



Features Of Blockchain

Decentralization

- No single authority controls the system
- Example: Sending money via Bitcoin without using a bank.



Transparency

- All participants can view and verify transactions.
- Example: In supply chain, everyone can track a product from factory to store.

Immutability

- Once recorded, data cannot be changed or deleted.
- Prevents fraud and manipulation.

Features Of Blockchain

Security

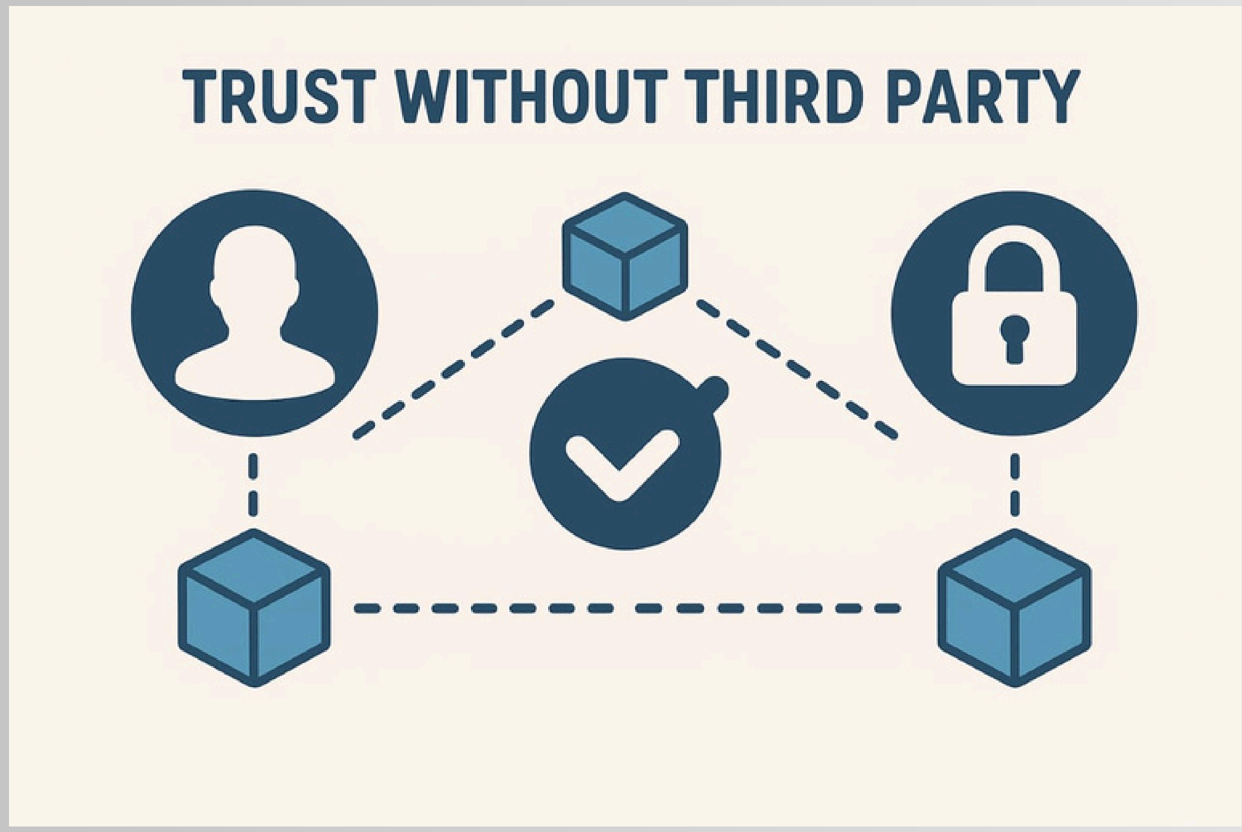
- Uses cryptography to protect data and identities.
- Difficult for hackers to alter records because data is stored on many computers.

Trust without Third Parties

- Blockchain builds trust through algorithms, not central authorities.
- Example: Smart contracts execute agreements automatically when conditions are met.

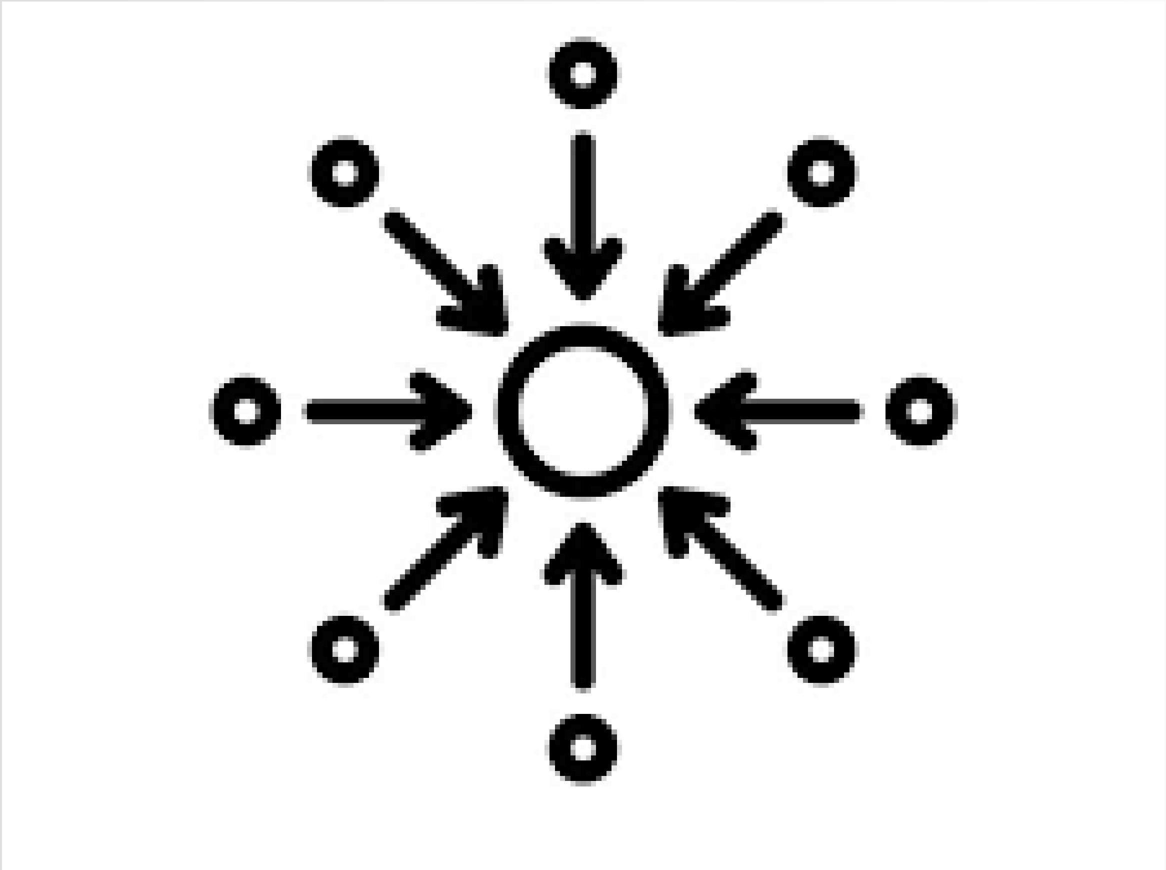
Consensus

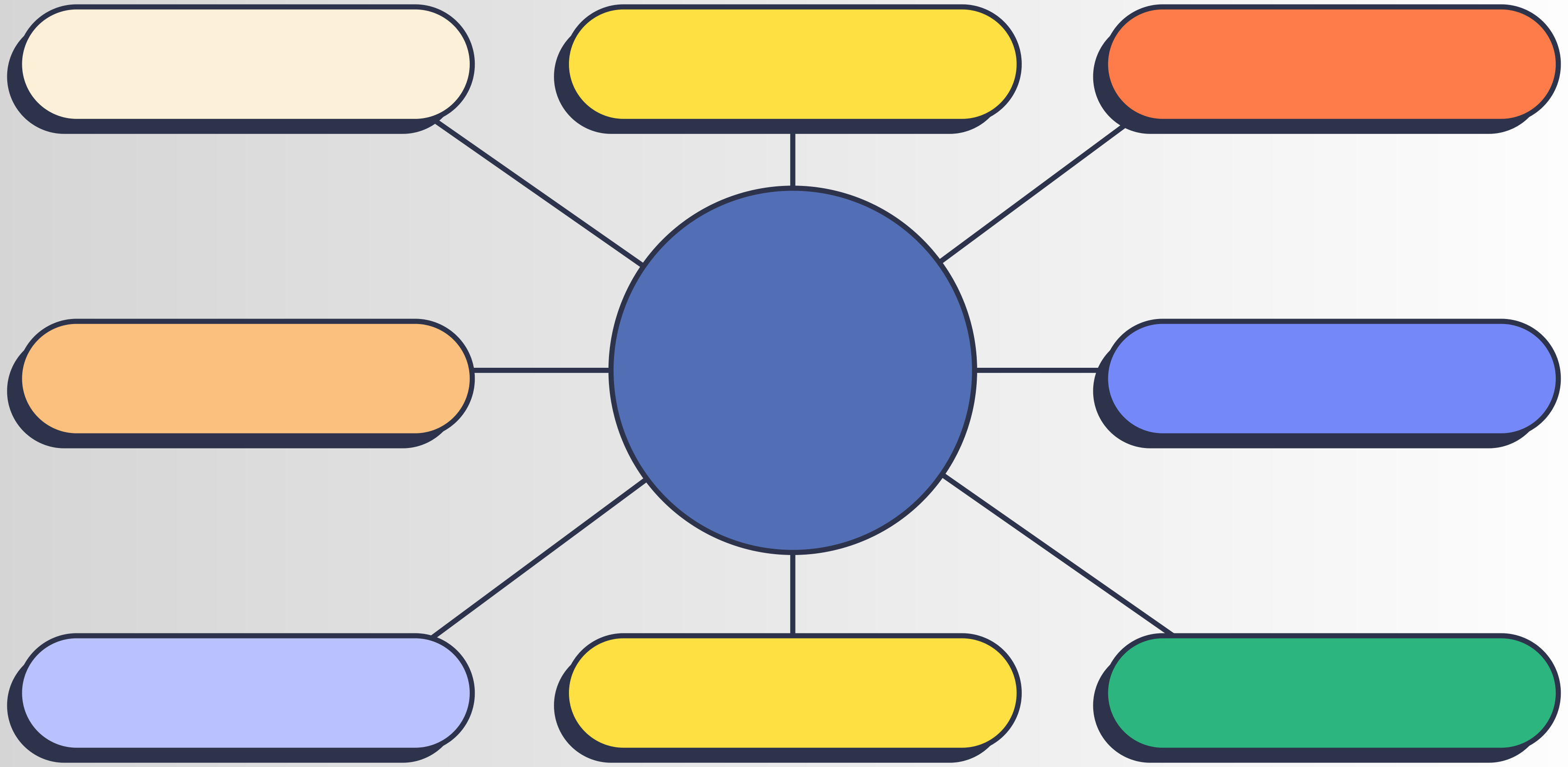
- Consensus help the network to make quick and unbiased decisions
- Maintains the security by keeping a record of all legitimate transactions.



Centralized network

- All the nodes are connected to a single point.
- This central node controls all the communications within the network.
- All the data is stored in one place
- All decision making power is only with few people
- Great for businesses as it allows them to control access to
- information and keep things organized and streamlined.





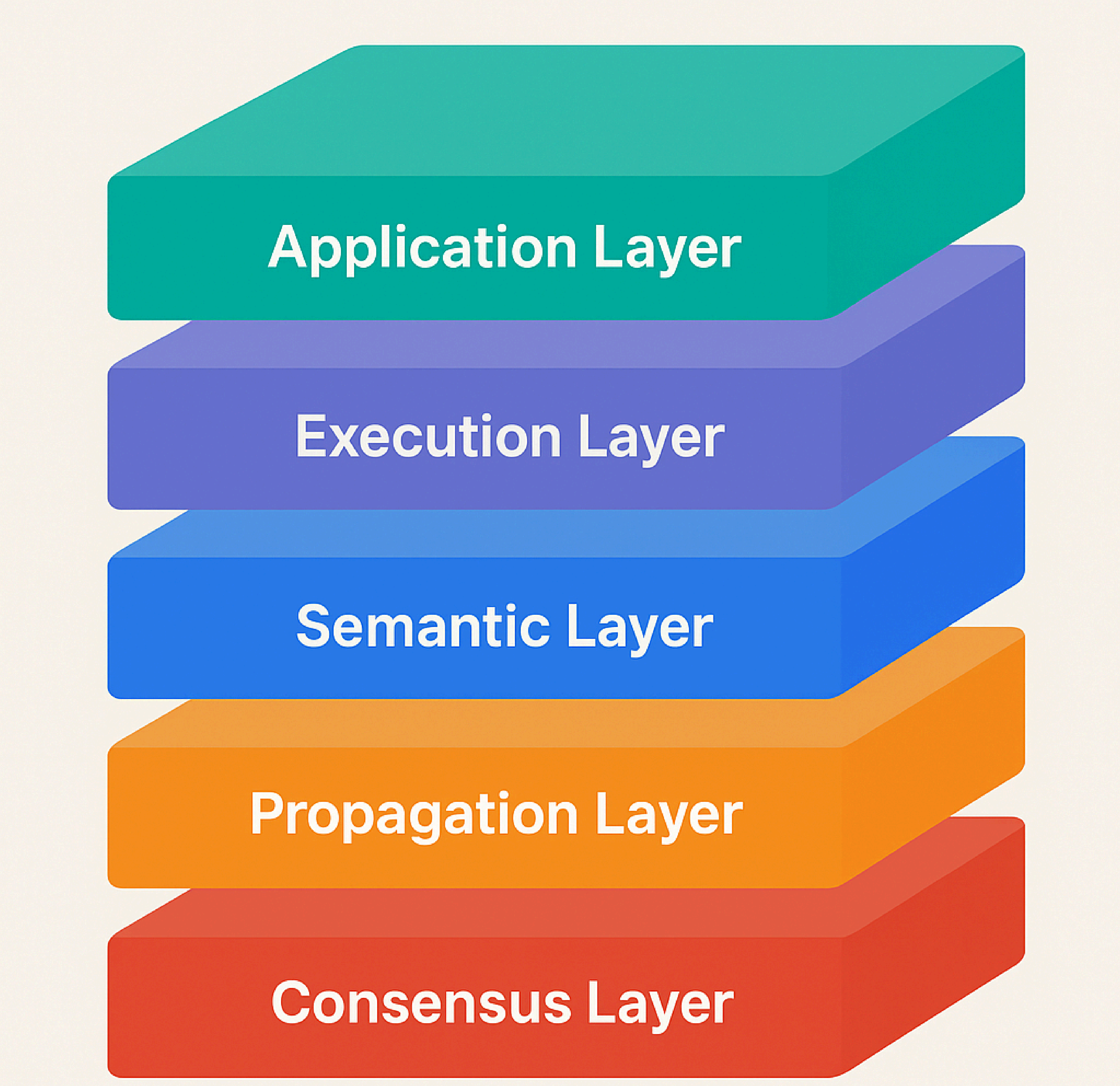
Decentralized network

- It is made up of multiple nodes that are all interconnected
- Distributes load among several machines
- No single person or group hold the authority
- Everybody in the network has the copy of ledger with them but no one can modify it on his/her own
- Difficult to control and manage, but it provides greater redundancy and fault tolerance.
- A decentralized network is more resistant to attacks.

Difference

Aspect	Centralized System	Decentralized System
Control	Single authority controls the entire system.	Control is distributed across multiple participants/nodes.
Decision Making	Faster, but decisions are made by one central authority.	Slower, as consensus is needed among participants.
Single Point of Failure	Yes – if the central authority/server fails, the whole system can collapse.	No – even if some nodes fail, the system continues to work.
Transparency	Low – data is controlled by the central authority.	High – data is visible to all participants (e.g., blockchain).
Security	More vulnerable to attacks on the central point.	More secure – difficult to attack all nodes simultaneously.
Scalability	Easier to scale vertically (by upgrading central servers).	Scales horizontally – adding more nodes improves resilience.
Trust	Requires trust in the central authority.	Trust is built into the system itself (via consensus, cryptography).
Examples	Banks, Social Media platforms (e.g., Facebook, Twitter).	Blockchain, Cryptocurrencies (e.g., Bitcoin, Ethereum).

Layers Of Blockchain



Application Layer

The Application Layer is the topmost layer of the blockchain architecture.
It is where users interact with the blockchain through applications, services, and smart contracts.

Functions of Application Layer:

- **User Interface –** Provides a platform for users to send/receive data or transactions.
- **Smart Contracts –** Contains self-executing programs that define rules for transactions.
- **Decentralized Applications (DApps) –** Runs services like finance apps, games, marketplaces on blockchain.
- **APIs & Wallets –** Allows developers and users to connect with blockchain easily.

Examples:

- **Wallet Apps:** MetaMask, Trust Wallet (for sending/receiving crypto)
- **DApps:** Uniswap (crypto exchange), Decentraland (virtual land), CryptoKitties (game)
- **Smart Contracts:** Automatically transferring ownership of digital art (NFT) when payment is received.

Execution Layer

The Execution Layer is the part of the blockchain where transactions and smart contract instructions are actually run and their results are produced.
It processes the rules defined in the Application Layer and ensures they are carried out correctly.

Functions of Execution Layer:

- **Transaction Execution** – Takes validated transactions and processes them according to blockchain rules.
- **Smart Contract Execution** – Runs the code of self-executing contracts to perform actions automatically.
- **State Management** – Updates the blockchain's state (e.g., account balances) after transactions are processed.
- **Interaction with Consensus Layer** – Passes results to the consensus layer for final verification and block addition.

Example:

- Suppose a smart contract says: "If Alice sends 1 ETH to Bob, transfer ownership of a digital artwork to Alice."
- The Execution Layer runs this code, updates balances, and changes the ownership record.

Semantic Layer

The Semantic Layer in blockchain is about the meaning and interpretation of data stored in the blockchain. It ensures that all participants understand the data in the same way.

Functions:

Common Understanding of Data

- Blockchain stores information (like transactions, contracts, votes, medical records, etc.).
- The semantic layer defines how this information should be structured, labeled, and interpreted, so everyone understands it the same way.
- Example: Date “10/08/2025” = always August 10, not October 8.

Data Interoperability

- Different blockchains or applications might use blockchain for different purposes (finance, healthcare, supply chain). The semantic layer makes sure these systems can talk to each other and share data meaningfully.
- Example: Supply chain (1 ton rice) = hospital (1000 kg rice) = trade (2204 lbs rice).

Smart Contracts Depend on It

- For smart contracts to work correctly, they need standardized meanings for inputs and outputs.
- The semantic layer ensures contracts run without misinterpretation of conditions.
- Example: A smart contract that refunds train tickets if "delay > 1 hour" needs everyone to agree on what exactly “1 hour delay” means.

Semantic Layer

Ensures Transparency and Trust

- Because all participants interpret data in the same way, it avoids disputes.
- This creates trust in decentralized systems, where there is no central authority.

Simple Example

- Imagine 3 countries using blockchain for international trade:
- Country A records "1 ton of rice"
- Country B records "1000 kg of rice"
- Country C records "2204 lbs of rice"

Without the semantic layer, this could cause confusion.
With the semantic layer, all systems agree that these mean the same amount, so contracts and payments happen smoothly.

Propogation Layer

The Propagation Layer is the part of blockchain that is responsible for sharing information across all the nodes (computers) in the network.

Whenever a transaction or a new block is created, it must be sent (propagated) to every node so that everyone has the same updated copy of the blockchain.

How it Works

- **Transaction Creation** – A user creates a transaction (e.g., Alice sends 1 Bitcoin to Bob).
- **Broadcast** – The transaction is broadcasted to nearby (connected) nodes.
- **Gossip Protocol** – Each node shares the transaction with its neighboring nodes, and they share with others, until all nodes receive it.
- **Verification** – Every node checks whether the transaction or block is valid (signature, balance, double-spending, etc.).
- **Consensus Layer** – After propagation, the consensus mechanism (like Proof of Work, Proof of Stake) decides if the block should be added to the blockchain.

Consensus Layer

Consensus Layer is one of the most important parts of a blockchain.
It decides how all the nodes (computers in the blockchain network) agree on a single version of the truth, such as:

- Which transactions are valid
- Which block should be added next to the chain

Without consensus, different nodes might have different records, and the blockchain would become unreliable.

Main Roles of Consensus Layer

- **Agreement –** All nodes must agree on the same blockchain state (same transactions, same blocks).
- **Trust without middleman –** Consensus removes the need for banks, governments, or any third-party to verify transactions.
- **Security –** Prevents fraud, double spending (same coin used twice), and malicious attacks.
- **Fairness –** Ensures no single person controls the system.

Consensus Mechanisms (Methods)

The Consensus Layer uses different algorithms to achieve agreement. Some popular ones:

Proof of Work (PoW)

- Computers solve hard puzzles.
- First one to solve adds the block.
- Example: Bitcoin.
- Secure but uses a lot of electricity.

Proof of Stake (PoS)

- Instead of solving puzzles, validators are chosen based on how many coins they “stake” (lock up).
- Example: Ethereum (after 2022 merge).
- Energy-efficient.

Delegated Proof of Stake (DPoS)

- People vote for a few delegates who add blocks.
- Example: EOS, TRON.

Practical Byzantine Fault Tolerance (PBFT)

- Nodes talk to each other and vote to confirm the next block.
- Example: Hyperledger Fabric.

Why is Blockchain important?

Decentralization
Removes the need for a single controlling authority

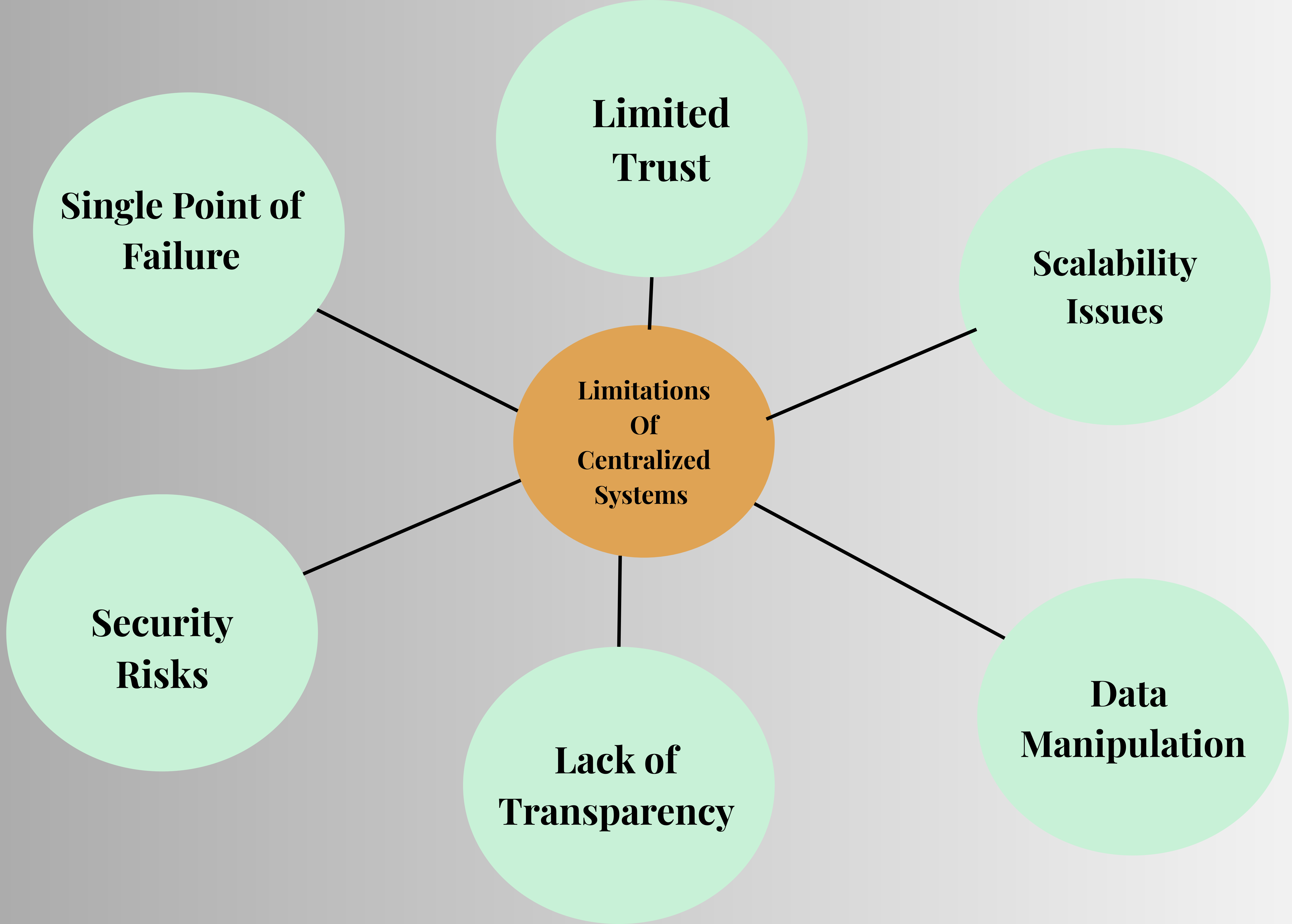
Trustless Transactions
Users don't need to rely on intermediaries like banks

Transparency
All transactions are visible to participants, reducing fraid

Immutability
Once recorded, data cannot be altered or deleted

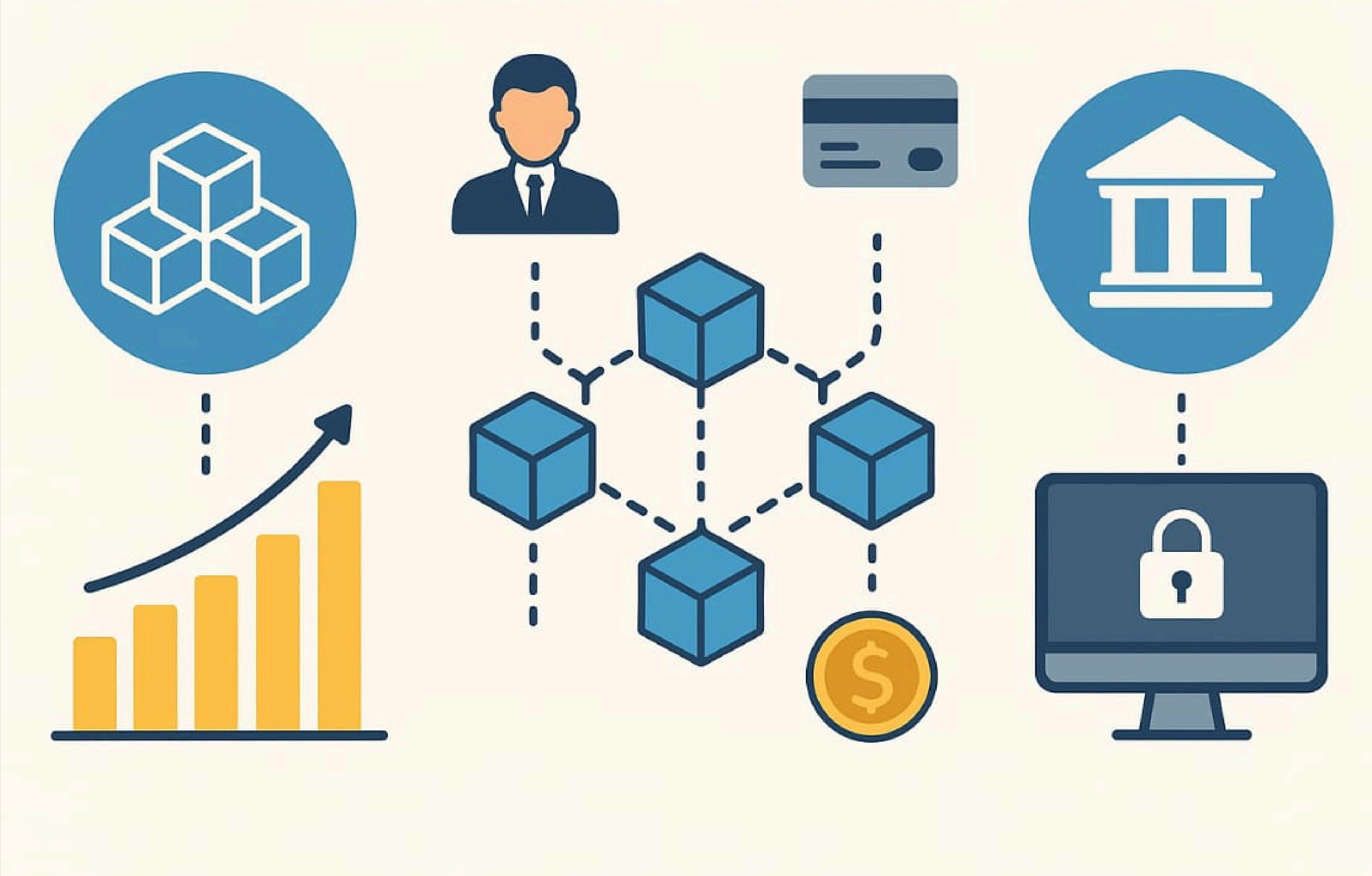
Security
Data is encrypted and nearly impossible to tamper with

Faster & Efficient
Eliminates delays caused by third parties


Limited Trust
Single Point of Failure
Scalability Issues
Limitations Of Centralized Systems
Security Risks
Lack of Transparency
Data Manipulation

Blockchain Adoption So Far.

- **Finance & Banking** – First adoption (Bitcoin 2009). Faster, cheaper money transfers. Example: Ripple (XRP) for cross-border payments.
- **Smart Contracts** – Ethereum (2015) enabled programs on blockchain. Example: DeFi apps for lending/trading without banks.
- **Supply Chain** – Tracks goods from factory to customer. Example: Walmart + IBM for food safety.
- **Healthcare** – Secures patient data & tracks medicines. Example: Anti-fake drug tracking.
- **Voting Systems** – Testing fraud-free, transparent e-voting. Example: Estonia's experiments.
- **Digital Identity** – People control their own IDs. Example: Aadhaar pilots with blockchain.



THANK YOU