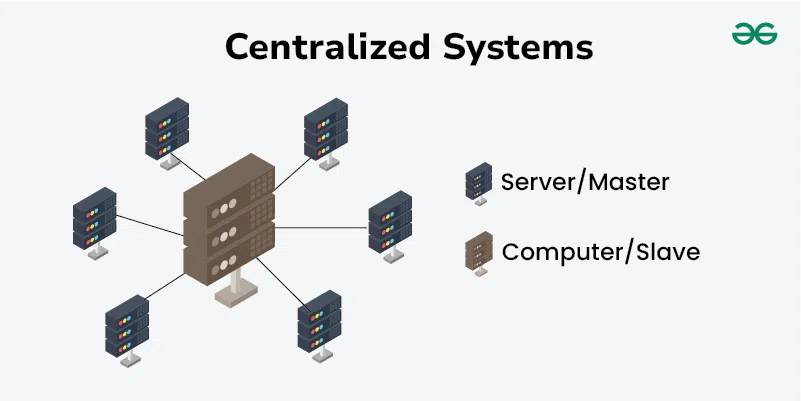
# TECHNICAL TERMS

1. Single Point of Failure (SPOF)
2. Single Point of Control
3. Resiliency
4. Scalability
5. Fault Tolerance
6. Centralized System
   * A **centralized system** is a system where a single central entity (server, database, or authority) controls all operations, decision-making, and data storage.
   * All users or nodes depend on this central entity for access and functionality.
   * **Characteristics of a Centralized System**
     + 1. Single Point of Control
          1. One central node manages the entire system.
          2. Users must communicate with the central server for processing.
       2. Single Point of Failure (SPOF)
          1. If the central server fails, the entire system stops working.
       3. Easy to Manage & Secure
          1. Since all data and processing are centralized, security policies are easier to enforce.
       4. Limited Scalability
          1. Adding more users increases load on the central server, leading to performance issues
       5. Faster Decision-Making
          1. Since all processing happens in one place, decisions are made quickly.
   * Real-Time Examples of Centralized Systems
     + 1. Traditional Banking System (SBI, ICICI, HDFC)
          1. All customer transactions go through a **central banking server**.
          2. If the bank’s **main server goes down**, no one can withdraw money or transfer funds.
          3. **Single point of failure**: If the central database is hacked, all user data is at risk.
       2. Railway Reservation System (IRCTC, Amtrak, Eurostar)
          1. All train ticket bookings, cancellations, and seat availability are managed by a **centralized server**
          2. If the IRCTC server is down, no one can book tickets.
       3. Government ID Systems (Aadhaar, Passport)
          1. All citizen data is stored in a **central government database**.
          2. If the server goes down, no one can access or verify their identity.
   * Advantages of Centralized Systems
     + 1. **Easy to Maintain & Secure** – Since all data is in one place, security and updates are simpler.
       2. **Fast Decision-Making** – Centralized processing speeds up operations.
       3. **Lower Cost** – Requires fewer resources compared to decentralized or distributed systems.
   * Disadvantages of Centralized Systems
     + 1. **Single Point of Failure** – If the central server fails, the entire system stops working.
       2. **Scalability Issues** – As demand grows, the system slows down.
       3. **Risk of Cyber Attacks** – If hackers breach the central system, they gain full access.
   * Summary
     1. **Centralized systems** are widely used but come with risks like **single points of failure** and **scalability issues**.
     2. **Modern systems** are shifting towards **decentralized** and **distributed** architectures for better **fault tolerance** and **scalability**.



1. Decentralized System
   * A **decentralized system** is a system where multiple nodes (servers, entities, or devices) , often spread across different locations , operate independently but still communicate with each other.
   * Unlike centralized systems, **there is no single controlling authority**; instead, decision-making is distributed across multiple nodes.
   * Instead of relying on **one central server**, multiple nodes manage data and processing, reducing the risk of failure and improving resilience.
   * Characteristics of a Decentralized System
     + 1. No Single Point of Failure (SPOF)
          1. If one node fails, the system continues to function.
       2. Multiple Authority Nodes
          1. Instead of a single master, multiple nodes handle decisions and processing.
       3. Better Security
          1. No central database means lower risk of hacking or data breaches.
       4. Scalability
          1. More nodes can be added to handle higher loads without major redesigns
       5. Latency Reduction
          1. Nodes operate independently, reducing dependency on a single data centre.
       6. Redundancy & Fault Tolerance
          1. Data is often replicated across multiple nodes, ensuring availability even if some nodes fail
   * Real-Time Examples of Decentralized Systems
     + 1. Blockchain (Bitcoin, Ethereum, Solana)
          1. **How it works?** Transactions are verified by multiple nodes instead of a central bank.
          2. **Why decentralized?** No single authority (like a bank) controls the network.
          3. **Example:** If one Bitcoin node fails, the network continues to function normally.
       2. Peer-to-Peer (P2P) Networks (BitTorrent, IPFS
          1. **How it works?** Files are shared across multiple computers instead of a central server.
          2. **Why decentralized?** Users (peers) directly exchange data without intermediaries
          3. **Example:** In BitTorrent, even if one peer goes offline, others continue sharing files.
   * Advantages of Decentralized Systems
     + 1. **Resilient & Fault-Tolerant** – If one node fails, others keep the system running.
       2. **More Privacy & Security** – No single entity controls all user data.
       3. **Scalable & Flexible** – Easy to add new nodes without central bottlenecks.
       4. **User Control & Transparency** – Users have more control over data and operations.
   * Disadvantages of Decentralized Systems
     + 1. **Complex to Manage** – Coordination between independent nodes can be difficult.
       2. **Higher Latency in Consensus-Based Systems** – Some decentralized systems (e.g., blockchain) take longer to reach consensus.
       3. **Security Risks in Open Networks** – If not designed well, attackers can manipulate some decentralized networks.

A diagram of several computer systems

AI-generated content may be incorrect.

1. Distributed System
   * A **distributed system** is a collection of multiple computers (or nodes) that work together to achieve a common goal.
   * These systems appear as a **single entity** to the end-user but are actually **spread across multiple locations**.
   * Unlike centralized systems, where everything is controlled by a single server, **distributed systems share resources, processing, and storage across multiple machines**.
   * Characteristics of a Distributed System
     + 1. Scalability
          1. Can handle more load by adding more nodes (computers).
          2. Example: Google adds more servers to process search queries faster.
       2. Fault Tolerance
          1. Even if one server fails, the system keeps working.
          2. Example: In Netflix’s distributed system, if one data centre goes down, another takes over
       3. Concurrency
          1. Multiple requests can be processed simultaneously.
          2. Example: Facebook handles millions of users at once.
       4. Location Transparency
          1. Users don’t need to know where data is stored or processed.
          2. Example: When using Google Drive, you don’t know which server is handling your request.
       5. High Availability
          1. The system is designed to be operational **24/7**.
          2. Example: Amazon’s e-commerce platform works worldwide without downtime.
   * Real-Time Examples of Distributed Systems
     + 1. Netflix, YouTube, Amazon Prime
          1. Videos are **stored on multiple servers** so users in different locations get the fastest response.
          2. If a server in the USA is slow, users in Europe get content from a closer data center.
       2. Content Delivery Networks (CDNs) - Cloudflare, Akamai
          1. Websites like YouTube and Netflix use CDNs to store content **closer to users** for faster streaming.
       3. Google Drive, Dropbox, OneDrive
          1. Data is replicated across multiple data centers so files are never lost.
       4. Cloud Computing Platforms : AWS, Microsoft Azure, Google Cloud
          1. Distributed cloud infrastructure runs applications for businesses worldwide.
       5. Microservices Architectures
          1. Architectures where applications are built as a collection of loosely coupled services.
   * Advantages of Distributed Systems
     + 1. **High Performance** – Handles millions of requests per second.
       2. **Fault Tolerance** – If one server fails, others take over automatically.
       3. **Better Load Balancing** – Distributes workload efficiently.
       4. **Cost-Effective** – Uses multiple cheaper machines instead of a single expensive one.
       5. **Data Replication** – Ensures that no data is lost.
   * Disadvantages of Distributed Systems
     + 1. **Complexity** – Managing multiple servers requires advanced networking and software.
       2. **Network Dependency** – Requires a strong internet connection to function efficiently
       3. **Synchronization Issues** – Keeping all nodes in sync is challenging.

A diagram of a distributed system

AI-generated content may be incorrect.

1. CDN - Content Delivery Network

# **CENTRALIZED VS DE-CENTRALIZED VS DISTRIBUTES SYSTEMS**

A blue and white logo

AI-generated content may be incorrect.

| **Feature** | **Centralized System 🏢** | **Decentralized System 🌍** | **Distributed System 🌎** |
| --- | --- | --- | --- |
| **Definition** | A single central server or authority controls everything. | Multiple independent entities manage different parts of the system. | A network of connected nodes works together to process tasks. |
| **Control** | Fully controlled by a **single entity** (company, government, etc.). | Control is **spread across multiple independent nodes**. | Nodes share control but **work together as one system**. |
| **Data Storage** | All data is stored in **one central server or database**. | Data is **replicated or split** among multiple independent nodes. | Data is **distributed across multiple locations** for redundancy. |
| **Failure Impact** | **High** – If the central server fails, the whole system goes down. | **Moderate** – Failure of one node doesn’t stop the entire system but may affect performance. | **Low** – If one or more nodes fail, the system continues functioning. |
| **Performance** | **Fast**, but bottlenecks occur due to overloading the central server. | **Varies** – Can be slow due to distributed decision-making. | **High** – Load is distributed, preventing slowdowns. |
| **Security Risks** | **High risk** – A single security breach can compromise the whole system. | **Moderate risk** – No single point of failure, but independent nodes may be vulnerable. | **Lower risk** – Redundancy and data replication provide better security. |
| **Scalability** | **Limited** – Expansion is difficult and requires upgrading the central system. | **Moderate** – Can scale by adding more independent nodes, but efficiency depends on coordination. | **Highly Scalable** – More nodes can be added without affecting performance. |
| **Data Consistency** | **High** – Since data is stored centrally, it is always consistent. | **Lower** – Different nodes may have **inconsistent data** due to lack of synchronization. | **High** – Distributed databases ensure data consistency. |
| **Maintenance & Cost** | **Expensive** – Requires **high infrastructure and IT support**. | **Moderate** – Lower initial costs but requires **complex governance**. | **Cost-Efficient** – Uses multiple standard servers instead of one expensive system. |
| **Example Systems** | **Traditional Banking (Central Banks, Visa, PayPal), Facebook, Google Search**. | **Blockchain (Bitcoin, Ethereum), Peer-to-Peer (P2P) Networks, Decentralized Social Media (Mastodon)**. | **Google Cloud, Netflix CDN, AWS, YouTube, Multiplayer Gaming Servers**. |
| **Best Use Cases** | **Small to Medium-Scale Applications** where centralized control is beneficial. | **Trustless Systems** like cryptocurrency, where users need control over their own data. | **Large-Scale Systems** requiring high performance, fault tolerance, and scalability. |

# **LOAD BALANCING**

# **CAP THEOREM**