Load Balancing

What is the need?

* If there is only 1 server Machine, is there any need of load Balancing?

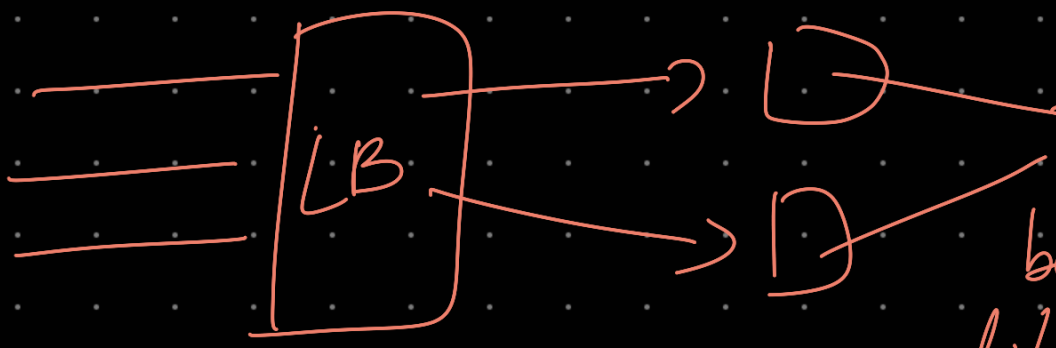
  → (NO.)

* In distributed server, need to distribute incoming traffic or requests across multiple servers to ensure no single server is overwhelmed.

  ⇓

  Increases Scalability, fault-tolerant system

  ⇓

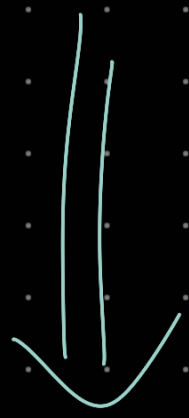


These could be two different machine or two different procss within some Machine managed using containization

will deep dive later.

Types of Load Balancers:

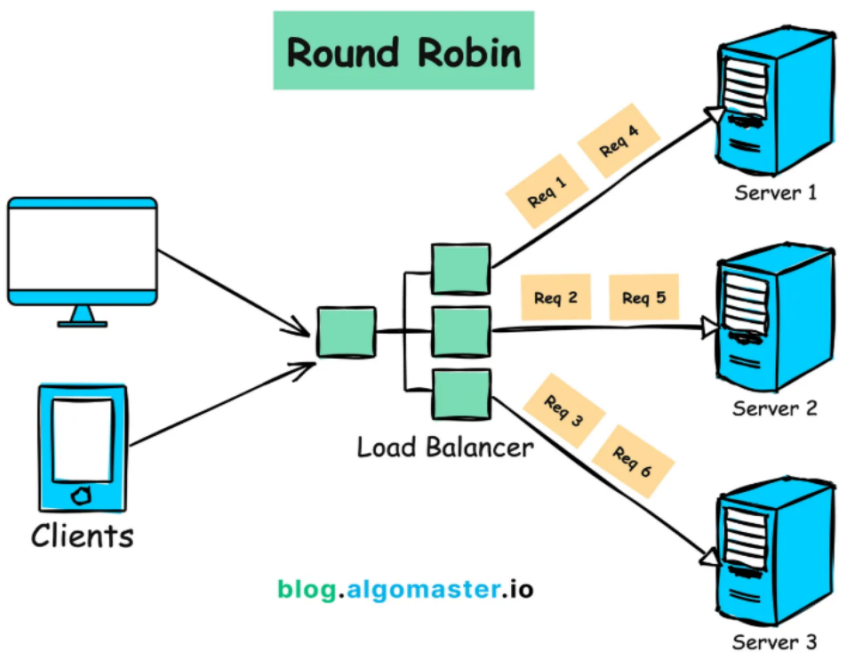
1. **Hardware Load Balancers:** Specialized physical devices for load balancing.
2. **Software Load Balancers:** Applications that perform load balancing (e.g., HAProxy, Nginx).
3. **Cloud-Based Load Balancers:** Services provided by cloud platforms (e.g., AWS Elastic Load Balancer, Azure Load Balancer).



Algorithm for Load Balancing

① Round-Robin

Algorithm 1: Round Robin



How it Works:

1. A request is sent to the first server in the list.
2. The next request is sent to the second server, and so on.
3. After the last server in the list, the algorithm loops back to the first server.

When to Use:

- When all servers have similar processing capabilities and are equally capable of handling requests.
- When simplicity and even distribution of load is more critical.

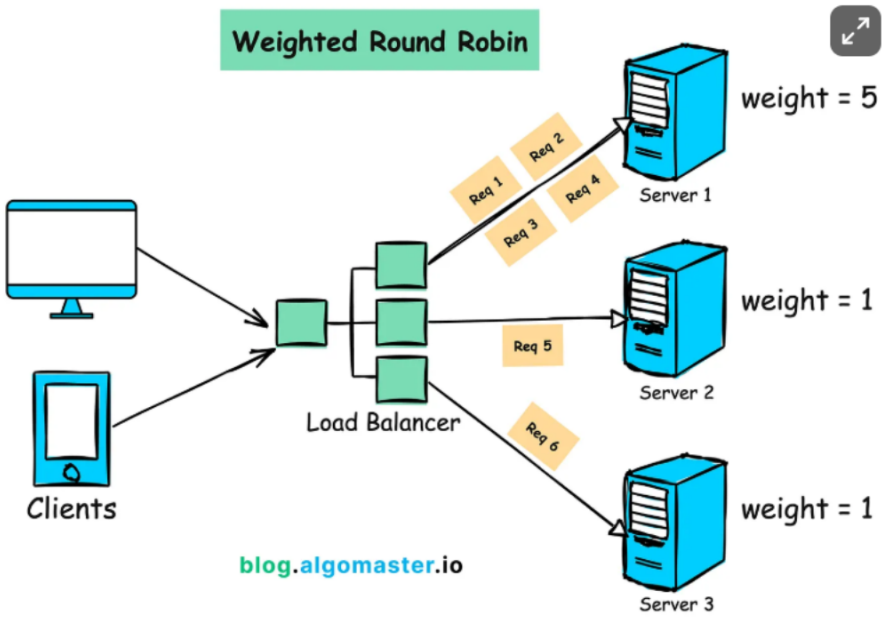
Benefits:

- Simple to implement and understand.
- Ensures even distribution of traffic.

Drawbacks:

- Does not consider server load or response time.
- Can lead to inefficiencies if servers have different processing capabilities.

② Weighted Round Robin



How it Works:

1. Each server is assigned a **weight** based on their processing power or available resources.
2. Servers with higher weights receive a proportionally larger share of incoming requests.

When to use:

- When servers have different processing capabilities or available resources.
- When you want to distribute the load based on the capacity of each server.

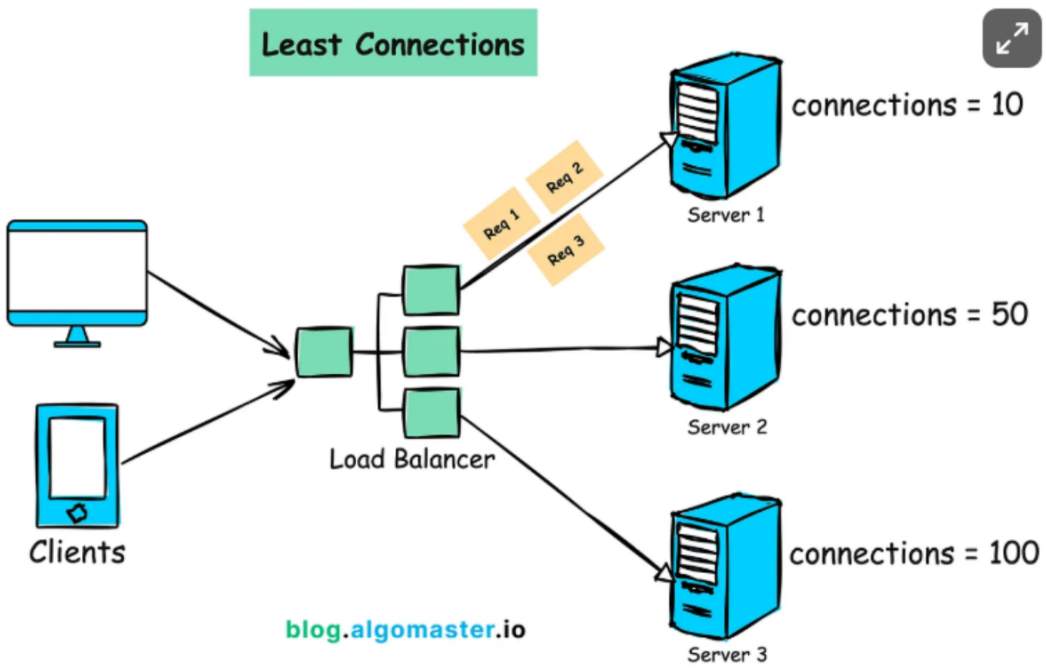
Benefits:

- Balances load according to server capacity.
- More efficient use of server resources.

Drawbacks:

- Slightly more complex to implement than simple Round Robin.
- Does not consider current server load or response time.

③ Least Connections



How it Works:

1. Monitor the **number of active connections** on each server.
2. Assigns incoming requests to the server with the least number of active connections.

When to use:

- When you want to distribute the load based on the current number of active connections.
- When servers have similar processing capabilities but may have different levels of concurrent connections.

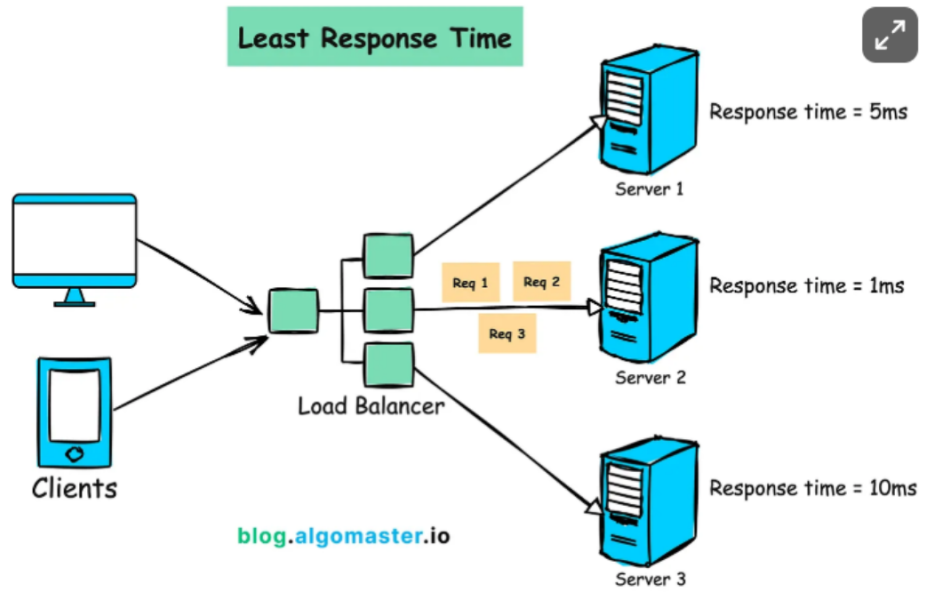
Benefits:

- Balances load more dynamically based on current server load.
- Helps prevent any server from becoming overloaded with a high number of active connections.

Drawbacks:

- May not be optimal if servers have different processing capabilities.
- Requires tracking active connections for each server.

④ Least Response time



How It Works:

- Monitors the response time of each server
- Assigns incoming requests to the server with the fastest response time.

When to Use:

- When you have servers with varying response times and want to route requests to the fastest server.

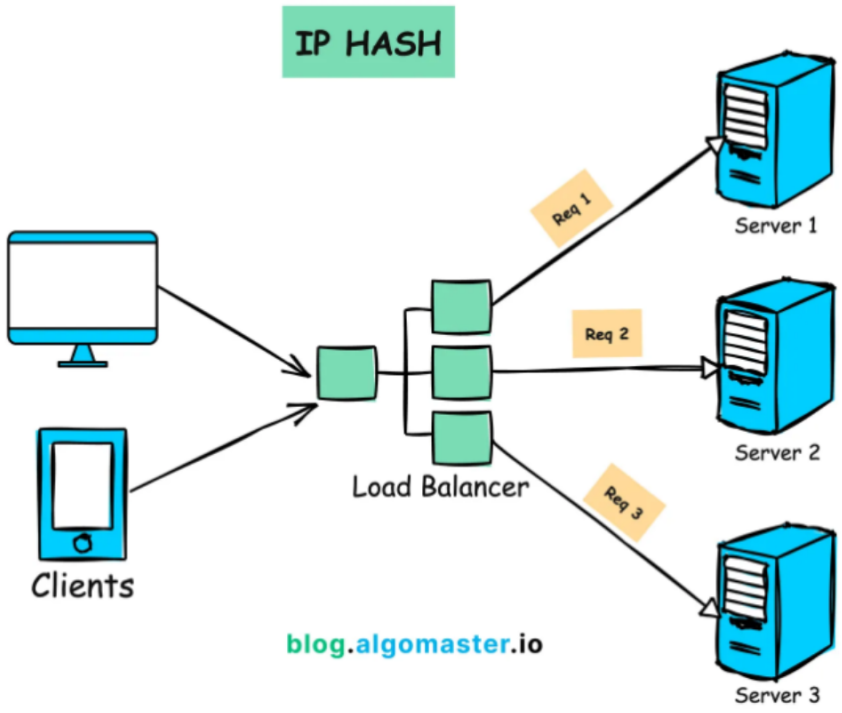
Benefits:

- Minimizes overall latency by selecting the server with the fastest response time.
- Can adapt dynamically to changes in server response times.
- Helps improve the user experience by providing quick responses.

Drawbacks:

- Requires accurate measurement of server response times, which can be challenging in distributed systems.
- May not consider other factors such as server load or connection count.

⑤ IP hash



How It Works:

- Calculates a hash value from the client's IP address and uses it to determine the server to route the request.

When to Use:

- When you need session persistence, as requests from the same client are always directed to the same server.

Benefits:

- Simple to implement.
- Useful for applications that require sticky sessions.

Drawbacks:

- Can lead to uneven load distribution if certain IP addresses generate more traffic than others.
- Lacks flexibility if a server goes down, as the hash mapping may need to be reconfigured.