# [System Design Key concepts]

## [Scalability]

↳ Property of system to handle growing Load by adding Resources to System

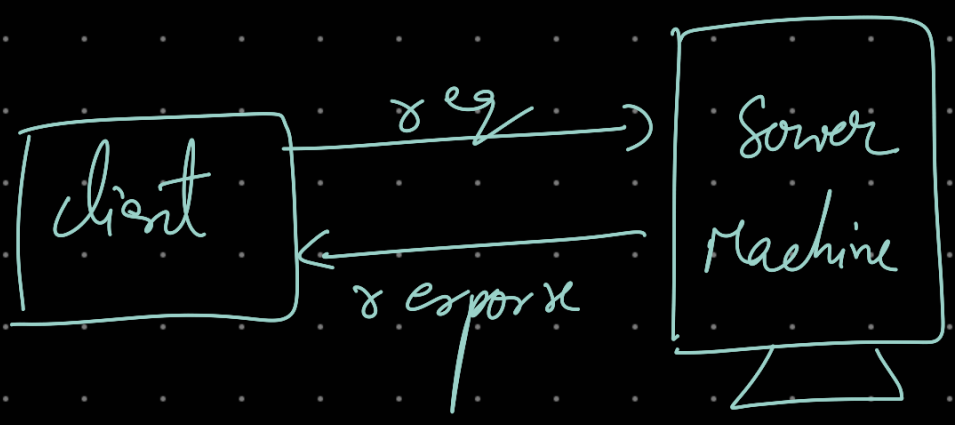
How Can System Grow?

- ↳ More Users
- ↳ More features
- ↳ More data
- ↳ More Complexity
- ↳ More Geographies

Vertical Scaling

Horizontal Scaling

E.g :-




No. of request is increasing

Vertical Scale

||

Add more RAM, CPU, HDD, and other resources to Same Machine

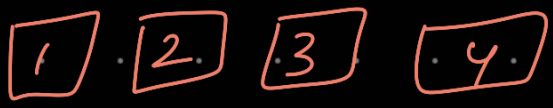
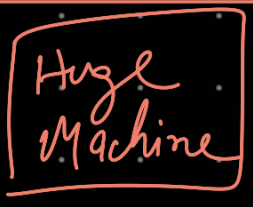
Horizontal Scale

||

Add more servers/Machine

11

Generally Managed by cloud providers like GCP, AWS, Azure etc.

| Horizontal | Vertical |
| --- | --- |
|   |   |
| ① Load balancing Required | ① N/A |
| ② Risilient, as Multiple server | ② Single point of failure |
| ③ Network calls (RPC) | ③ Inter process communication |
| ④ Data inconsistency | ④ Consistent |
| ⑤ Scale well | ⑤ Hardware limitation |

Both are used in Real world

⇓

Vertical scale initially

⇓

When it reaches limit

⇓

Horizontally scale

[Availability]

↳ Proportion of time a system is operational and accessible when required.

$$\text{Availability} = \frac{(\text{Uptime})}{(\text{Uptime} + \text{Downtime})}$$

## Availability Tiers

Availability is often expressed in "nines". The higher the availability, the less downtime there is.

| Availability % | Downtime per year | Commonly referred as |
|---|---|---|
| 99% | 3.65 days | "Two nines" |
| 99.9% | 8.76 hours | "Three nines" |
| 99.99% | 52.56 minutes | "Four nines" |
| 99.999% | 5.26 minutes | "Five nines" |
| 99.9999% | 31.5 seconds | "Six nines" |

Each additional "nine" represents an order of magnitude improvement in availability.

> **Example:** 99.99% availability represents a **10-fold** improvement in uptime compared to 99.9%.

# How to Improve Scalability

## 1. Redundancy

- Server Redundancy
- Database Redundancy
- Geographic Redundancy

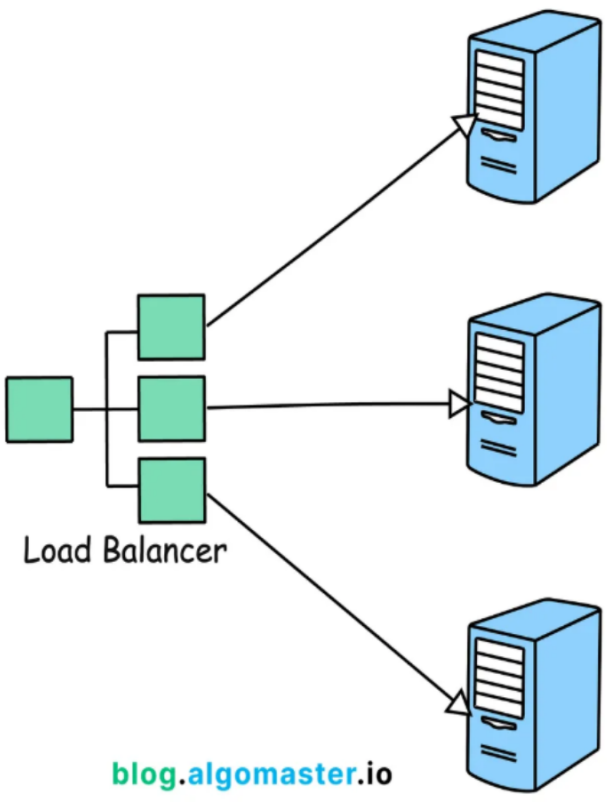
**Techniques:**

- **Server Redundancy**: Deploying multiple servers to handle requests, ensuring that if one server fails, others can continue to provide service.
- **Database Redundancy**: Creating a replica database that can take over if the primary database fails.
- **Geographic Redundancy**: Distributing resources across multiple geographic locations to mitigate the impact of regional failures.

# 2. Load Balancing

## 2. Load Balancing

Load balancing distributes incoming network traffic across multiple servers to ensure no single server becomes a bottleneck, enhancing both performance and availability.



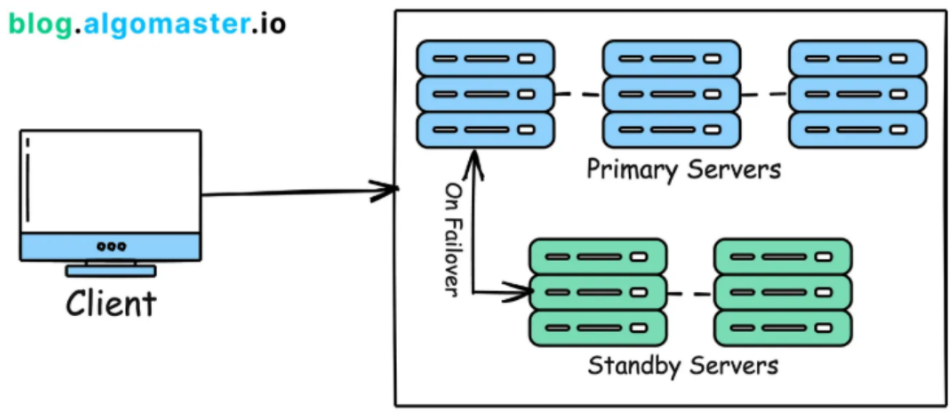

### Techniques:

- **Hardware Load Balancers**: Physical devices that distribute traffic based on pre-configured rules.
- **Software Load Balancers**: Software solutions that manage traffic distribution, such as HAProxy, Nginx, or cloud-based solutions like AWS Elastic Load

# 3. Failover Mechanisms

## 3. Failover Mechanisms

Failover mechanisms automatically switch to a redundant system when a failure is detected.

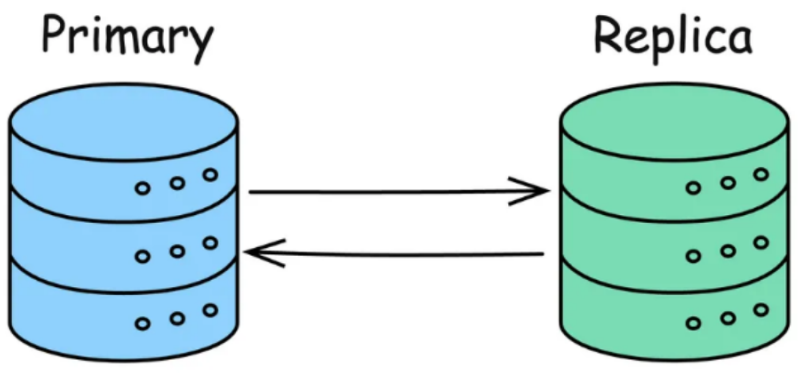



### Techniques:

- **Active-Passive Failover**: A primary active component is backed by a passive standby component that takes over upon failure.
- **Active-Active Failover**: All components are active and share the load. If one fails, the remaining components continue to handle the load seamlessly.

# 4. Data Replication

Data replication involves copying data from one location to another to ensure that data is available even if one location fails.



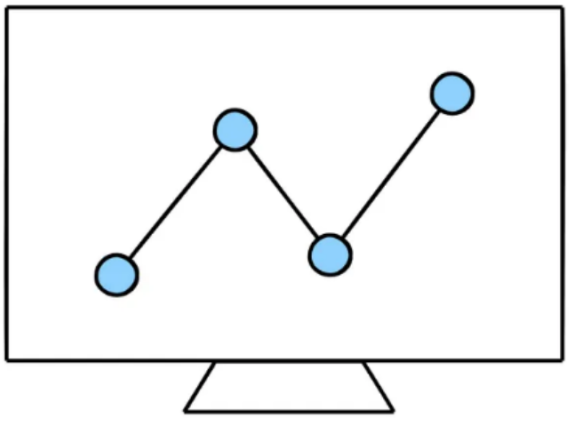

**Techniques:**

- **Synchronous Replication**: Data is replicated in real-time to ensure consistency across locations.
- **Asynchronous Replication**: Data is replicated with a delay, which can be more efficient but may result in slight data inconsistencies.

# 5. Monitoring Alerts

## 5. Monitoring and Alerts

Continuous health monitoring involves checking the status of system components to detect failures early and trigger alerts for immediate action.



### Techniques:

- **Heartbeat Signals**: Regular signals sent between components to check their status.
- **Health Checks**: Automated scripts or tools that perform regular health checks on components.
- **Alerting Systems**: Tools like PagerDuty or OpsGenie that notify administrators of detected issues.