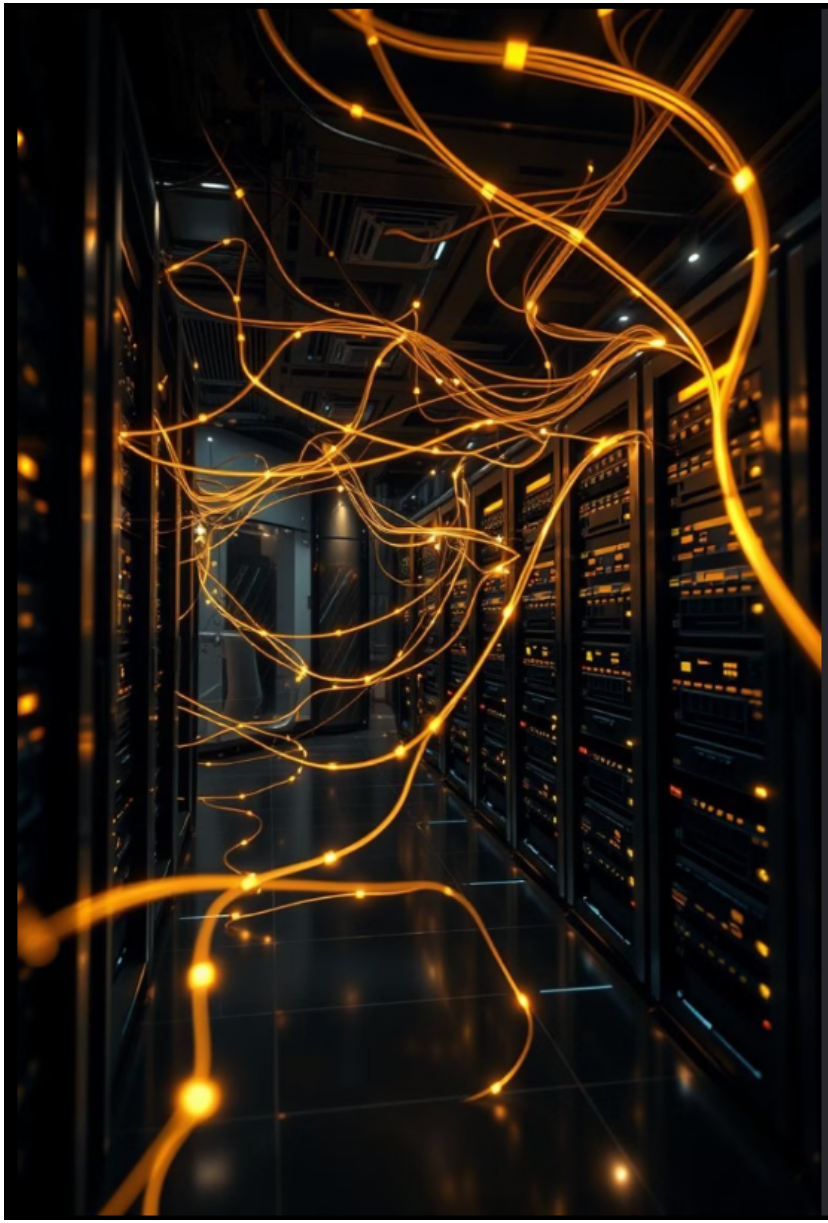


System Design: What is Scalability?

Scalability is a system's ability to handle increasing load by adding resources. It means adapting to more users, data, or features without performance compromise. This prevents bandwidth and latency issues.

Hareesh Rajendran

How Can a System Grow?

1. **User Base Growth**

   More users mean increased requests. A social media platform is a prime example.

2. **Feature Expansion**

   Adding features broadens system capabilities, like new payment options on an e-commerce site.

3. **Data Volume Increase**

   More user activity generates more data. A video streaming service stores vast content.

4. **Complexity Evolution**

   Architecture evolves, increasing system complexity. A simple app becomes a multi-component system.

5. **Geographic Reach**

   Expanding to new regions increases system reach. An e-commerce platform goes international.

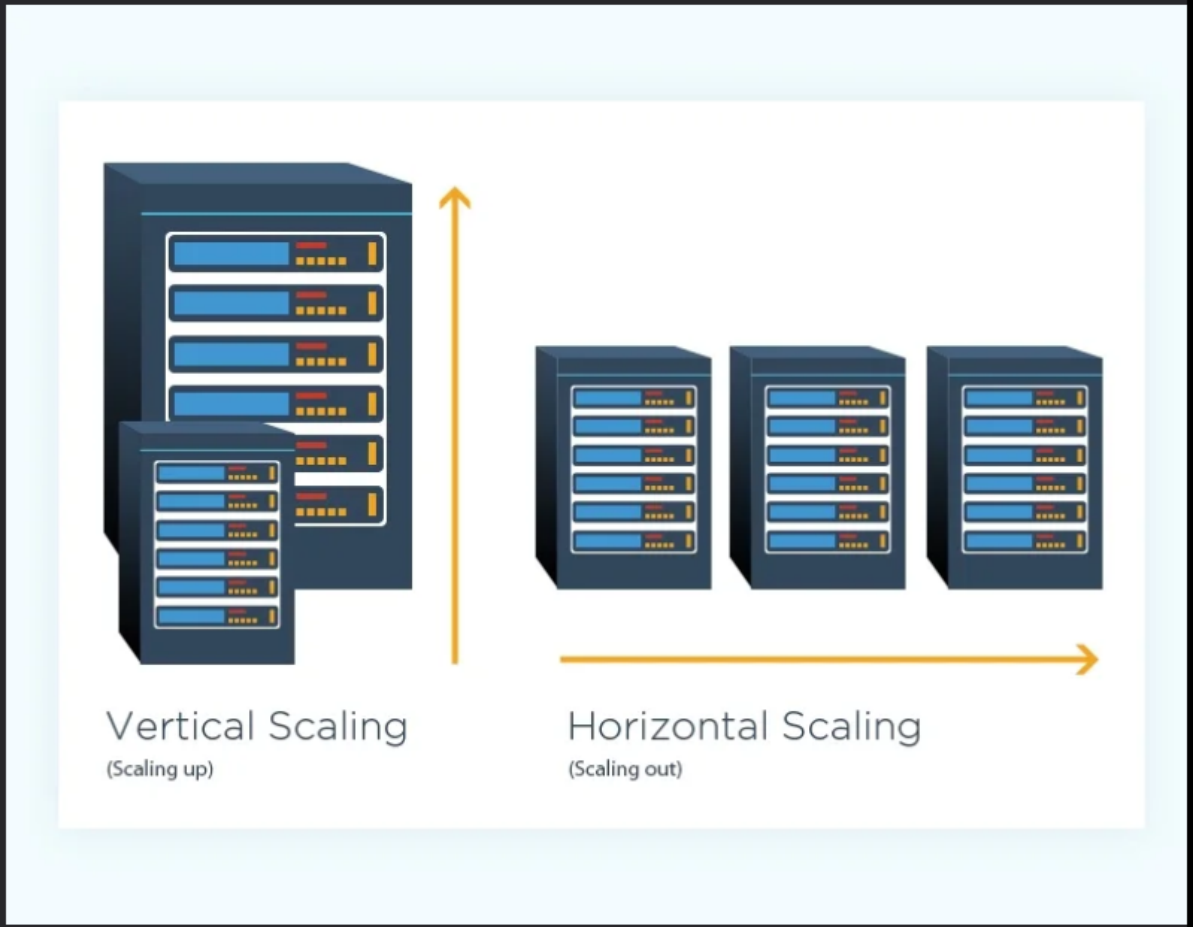
How to Scale a System?

Vertical Scaling (Scale Up)

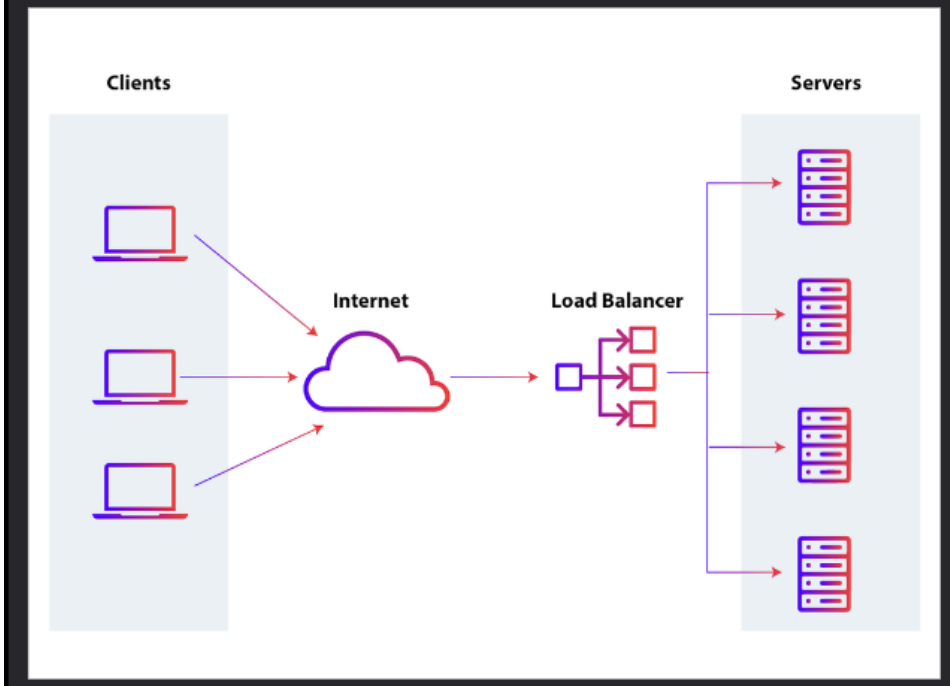
Upgrade existing servers with more RAM, faster CPUs, or additional storage. This is suitable for simpler architectures but has physical limits.

Horizontal Scaling (Scale Out)

Add more machines to distribute the workload. Netflix, for example, adds servers to handle increased data traffic efficiently.



Load Balancing



Incoming Traffic

User requests arrive at the load balancer.

Distribution

The load balancer routes requests.

Multiple Servers

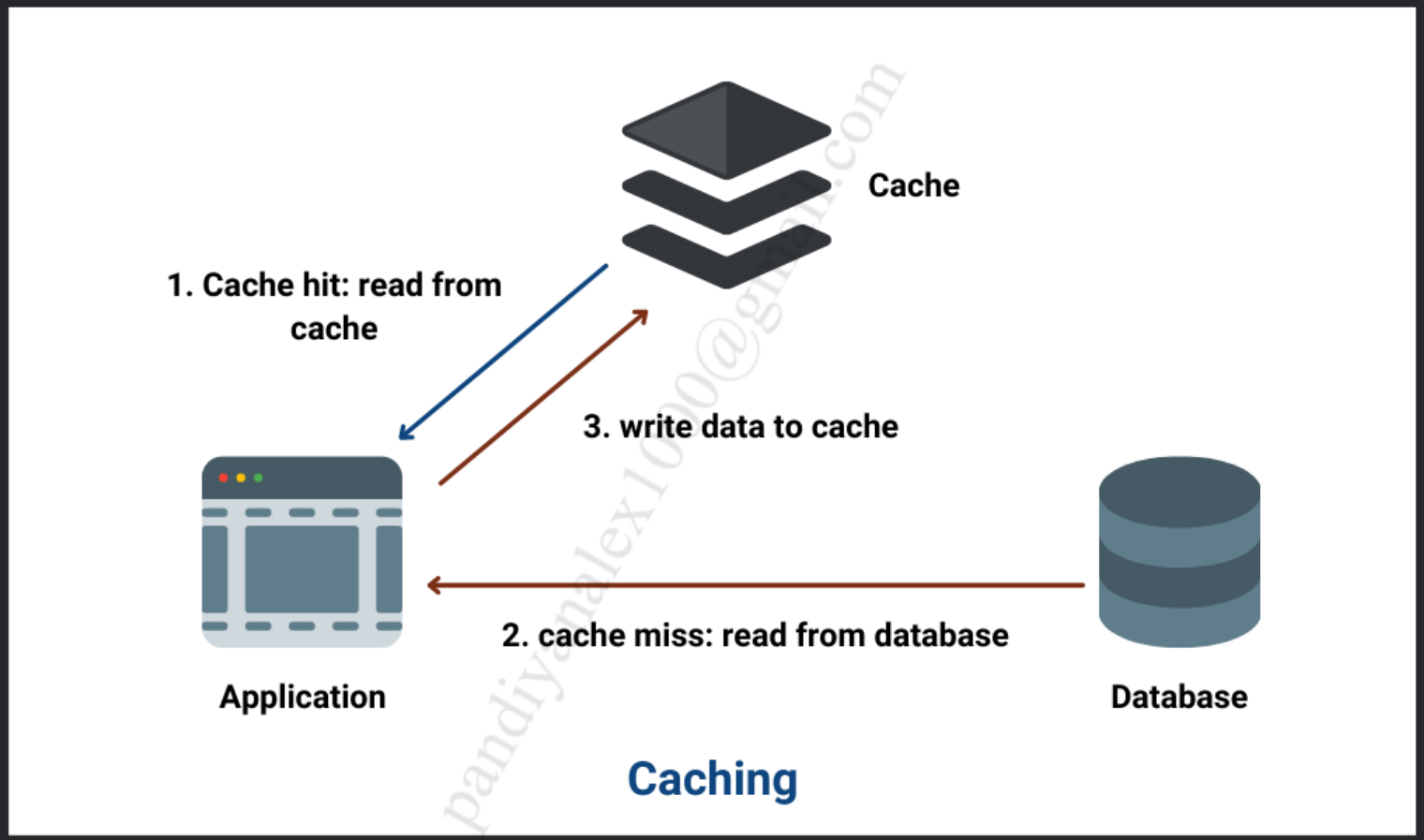
3

Traffic is spread across available servers.

Optimized Performance

Avoids overload and ensures responsiveness.

Caching


Cache
1. Cache hit: read from cache
3. write data to cache
2. cache miss: read from database
Application
Database
Caching

Caching

Data Request

A user requests data from the system.

Famous Scaling Techniques

Redis Cache or In-memory Cache

Check Cache

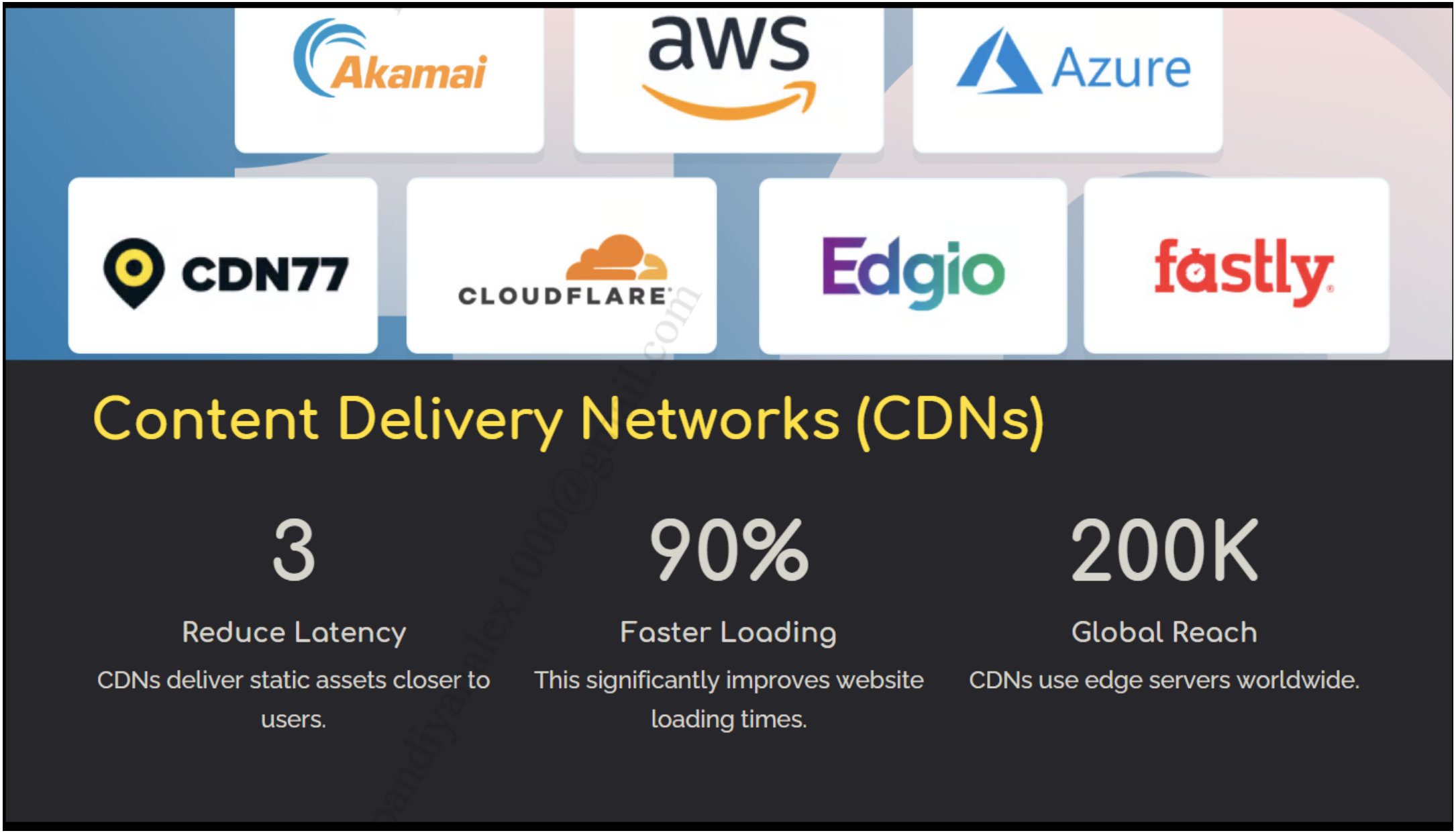
The system first checks the cache for the data.

Fast Retrieval

If found, data is served quickly from cache.

Retrieve & Store

If not, retrieve from main storage and cache it.


Akamai
aws
Azure
CDN77
CLOUDFLARE
Edgio
fastly
Content Delivery Networks (CDNs)
3
Reduce Latency
CDNs deliver static assets closer to users.
90%
Faster Loading
This significantly improves website loading times.
200K
Global Reach
CDNs use edge servers worldwide.

Partitioning

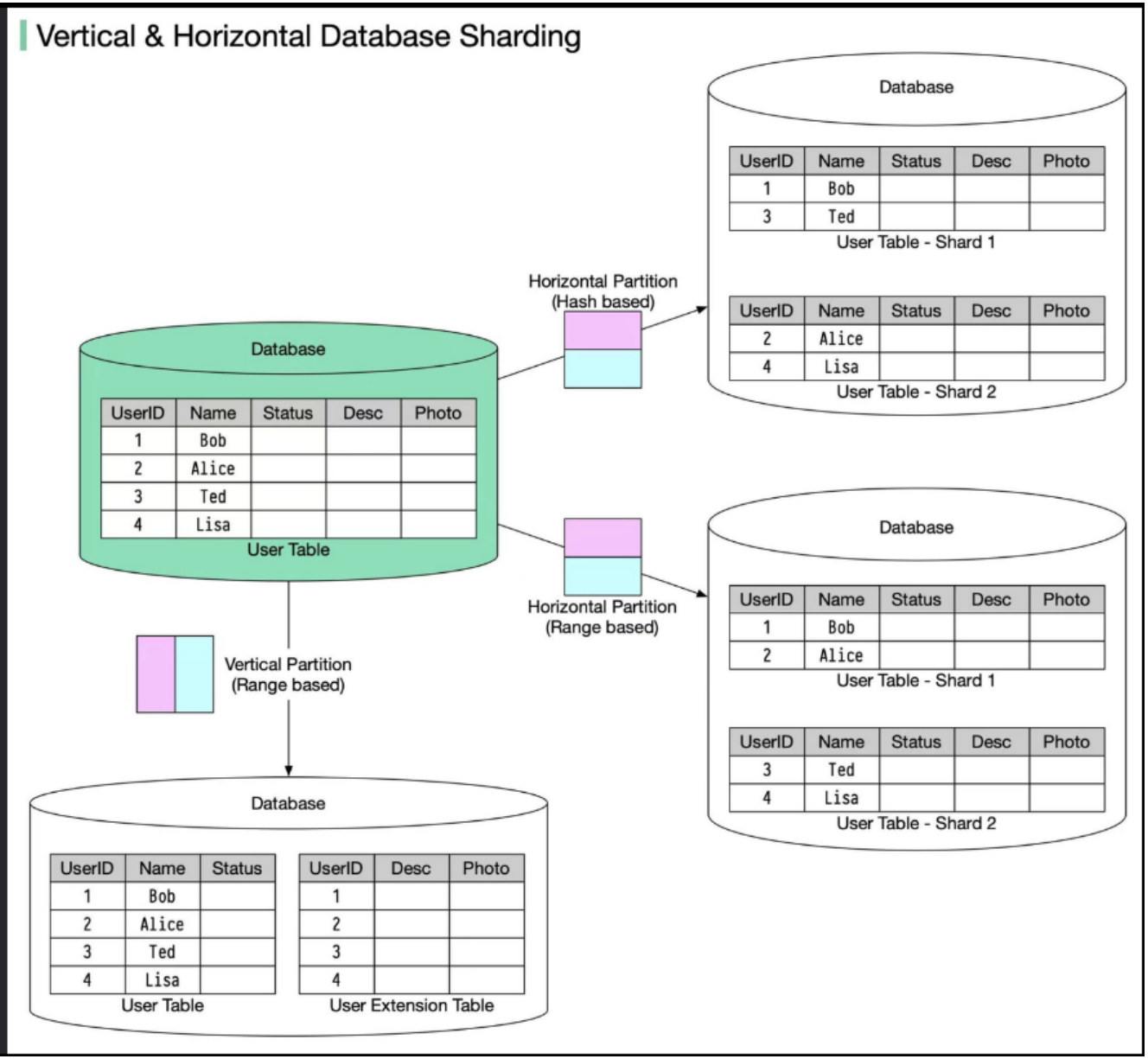
Horizontal Partitioning (Sharding)

Divides a database table's rows into smaller, more manageable parts. Each part is stored on a separate server.

Vertical Partitioning

Splits a database table's columns into multiple tables. This is often based on access patterns or data types.

Partitioning



CAP Theorem

The CAP theorem, also known as Brewer's theorem, states that any distributed data store can only guarantee two out of three desirable properties: Consistency, Availability, and Partition Tolerance. This fundamental principle highlights the inherent trade-offs in designing distributed systems

Consistency (C)

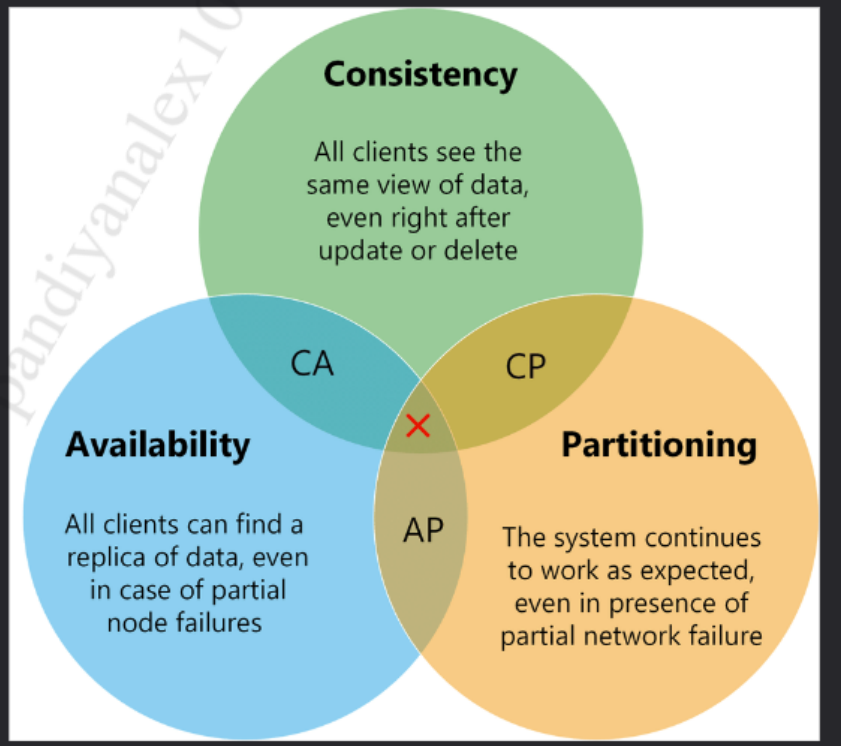
Every read operation receives the most recent write or an error. In a consistent system, all nodes see the same data at the same time

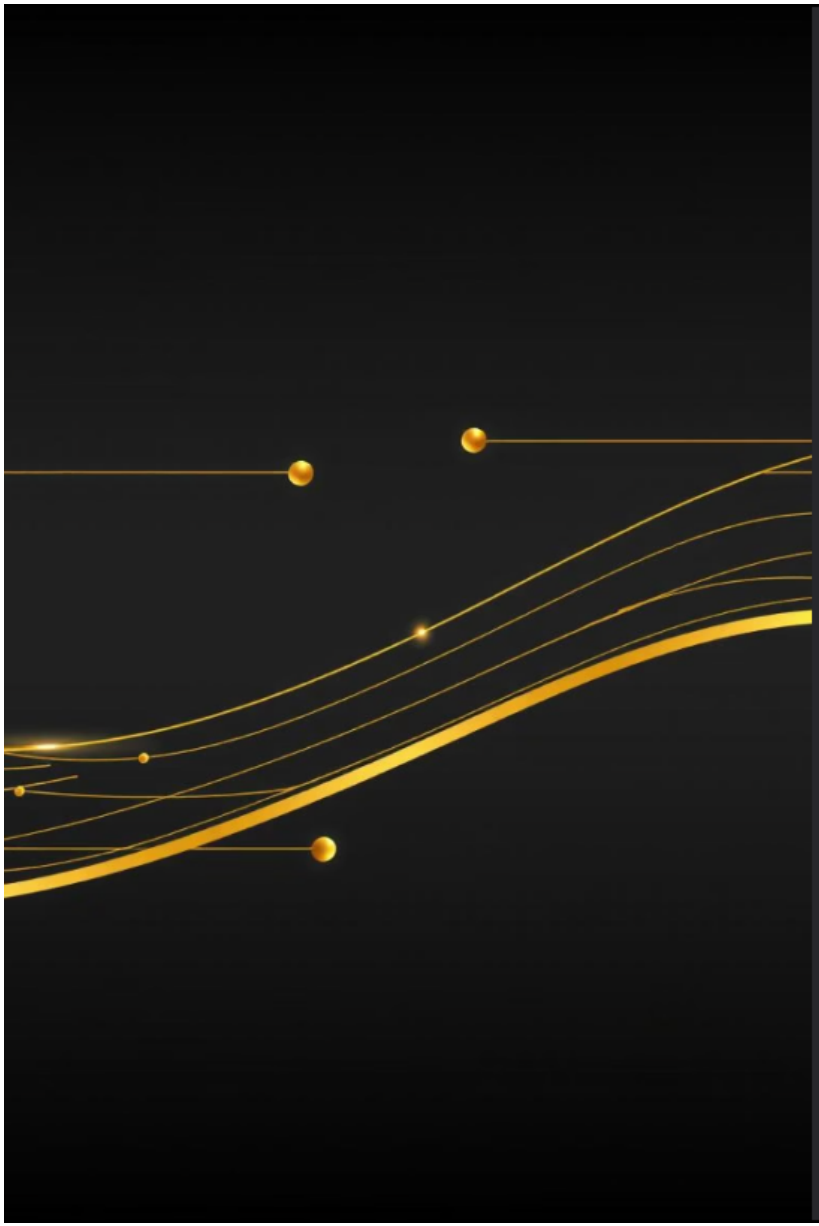
Availability (A)

Every request receives a response, even if it's not the most recent data. The system remains operational even if some nodes are down.

Partition Tolerance

The system continues to operate despite network partitions (communication breakdowns between nodes).





Asynchronous Communication

kafka

RabbitMQ

User Action

A request is initiated by the user.

Queue Task - MQ (RabbitMQ or Kafka)

Non-critical tasks are sent to a background queue.

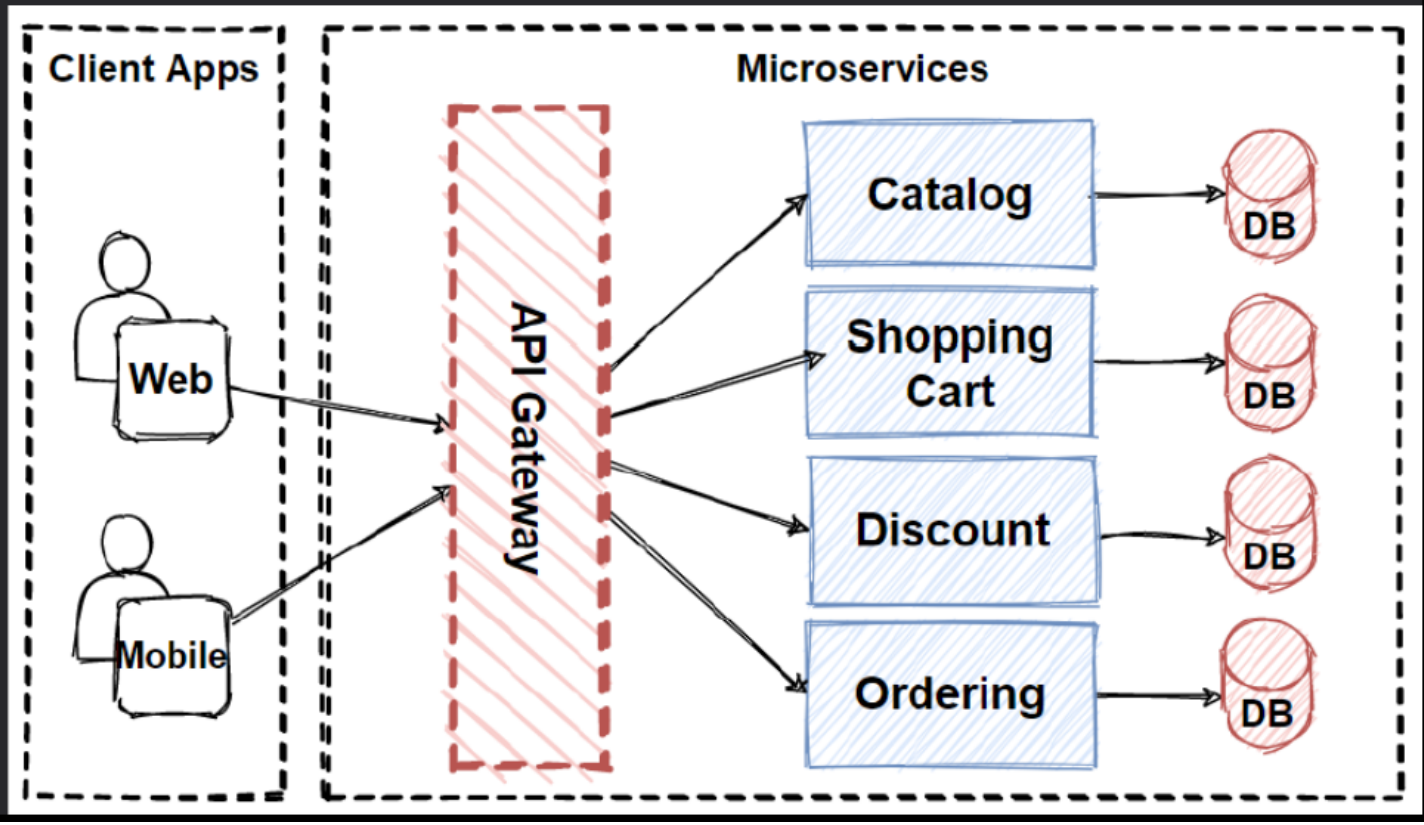
Immediate Response

The system provides an instant UI response.

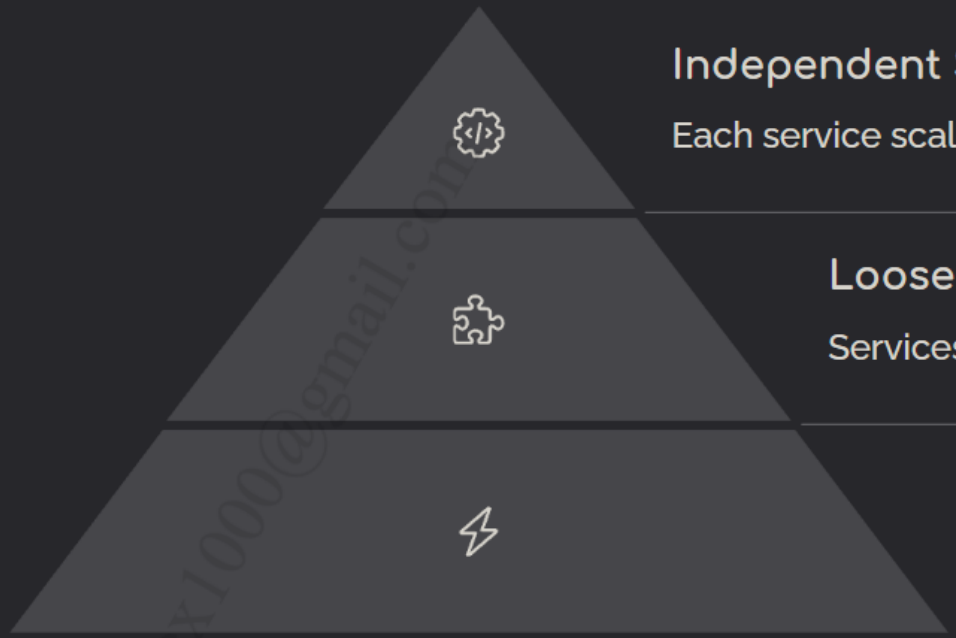
Background Processing

Tasks are processed independently, without blocking the main flow.

Microservices Architecture



Microservices Architecture



Independent Scaling

Each service scales according to its needs.

Loose Coupling

Services can be developed and deployed independently.

Enhanced Agility

Faster development and deployment cycles.

Auto-Scaling and Multi-region Deployment

Traffic Spike

System detects a surge in demand.

Add Resources

Auto-scaling automatically provisions more servers.

Global Deployment

Application deployed across multiple geographic regions.

Seamless Experience

Users receive low-latency and highly available service.