Concepts	Technologies	Application
Stateless & Stateful	Kubernetes	
Virtualization v/s Containers		
Orchestration		
DevOps		
CI/CD		

Docker

Dockerfile

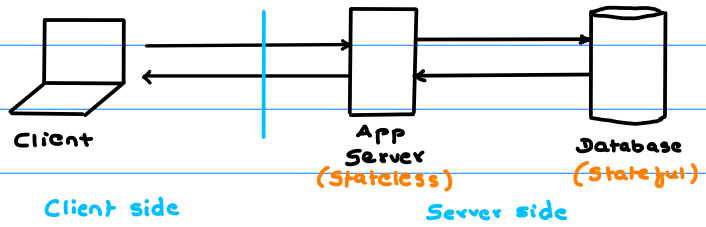
Docker registry

Multi-host

Cross-host networking

State/Volume

1. Image building
2. Container ↔ Container networking
3. Volume management

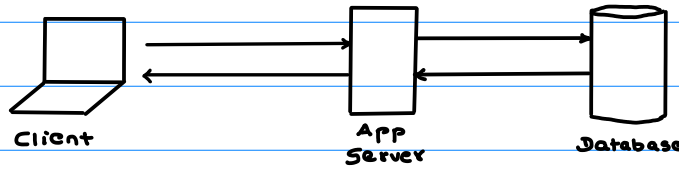


Anytime, anywhere, any device

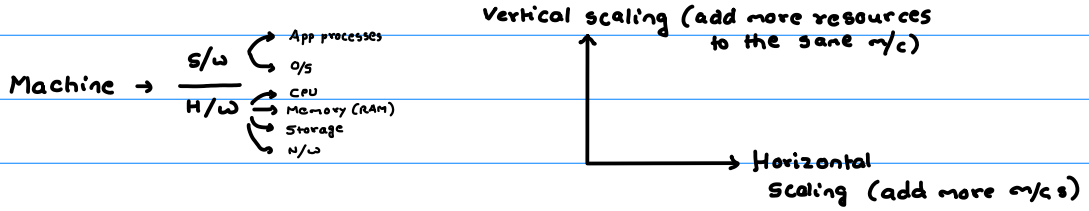
State, Stateless, Stateful

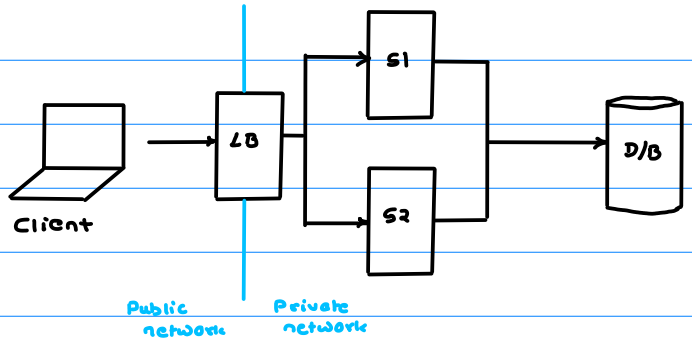
	Scalability	Failure Handling
Stateless	LB + Replicas	Replace replica
Stateful	Sharding	Replication

Stateless Scalability



100 Req/s



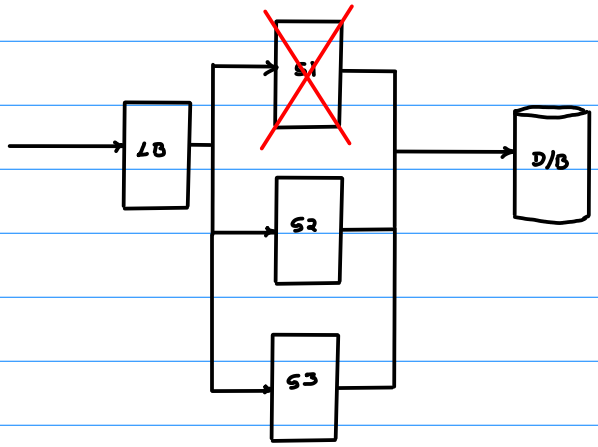


Load balancer

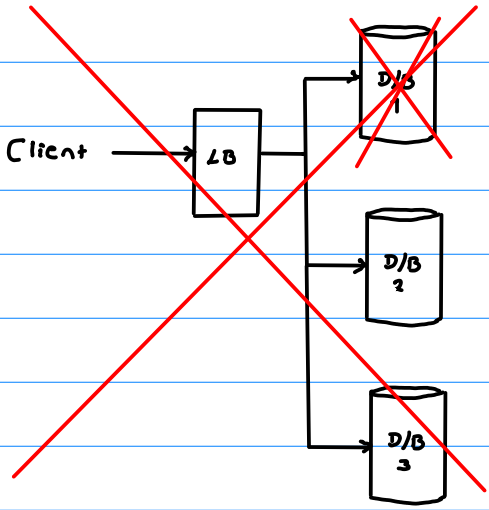
1. Ingress
2. Load balancing / Round Robin / Reverse Proxy
3. Failure management

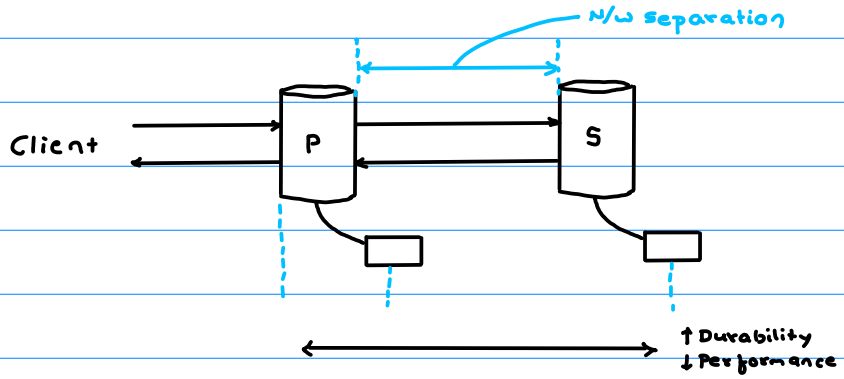
Health check endpoint

/ping → 200 OK



Stateful scalability

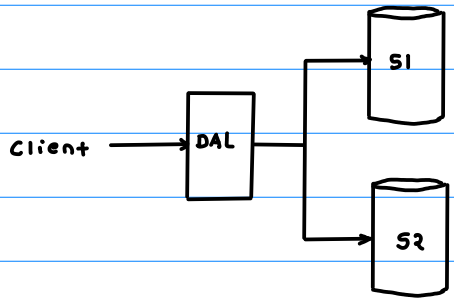




CRUD
- Read
- Write

Types of failure

1. M/c failure → S/w, H/w → Multiple m/c
2. Power, Fire → Multiple A/z
3. Cyclone, Earthquake → Multiple region



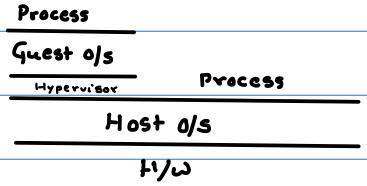
Shard key → Consistent Hashing → 0/1

Virtualization

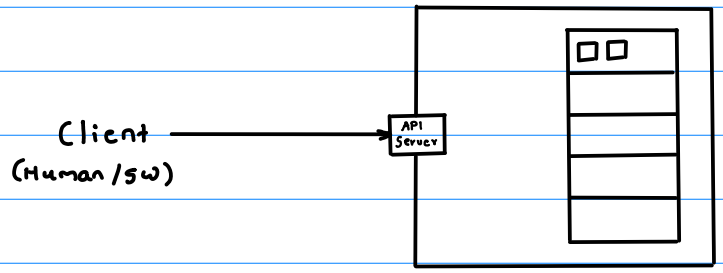
Virtual Machine → S/W (Process, O/S) / H/W (CPU, RAM, Storage, N/W)

↳ Not real

Real Machine



Isolation of resources



AWS → 2006

Google Cloud → 2008

ELB, RDS → 2009

Cloudformation → 2010

Azure → 2010

IAAS

PAAS

IAC

Reduced admin

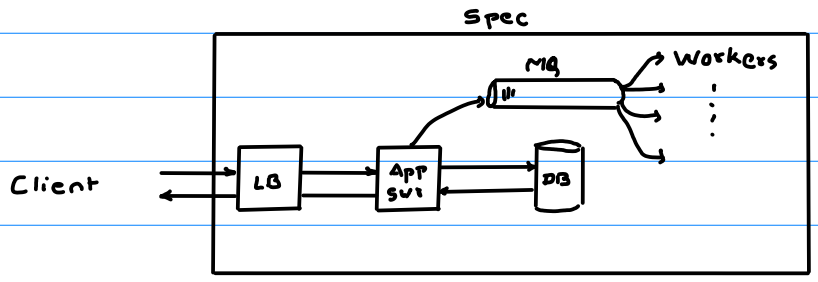
Increased automation

Public

Private

Hybrid

Multi-cloud



App Servers

Databases

Load balancers

Caching Servers

Message queues

(IAC) Spec → Orchestrator → Create / Operate

Vendor lock-in

Kubernetes → FOSS

↳ Container based orchestration

↳ Google Kubernetes Engine

Amazon Kubernetes Service

Azure —"— —"—

Docker

RedHat / IBM

VMware / Pivotal

Containers

Process

O/S

H/W

cgroups
namespaces
chroot

Container v/s VM

1. Resource isolation
2. VMs need more resources
   Containers are light-weight
3. VMs are more secure
   Containers are not as secure

Kubernetes