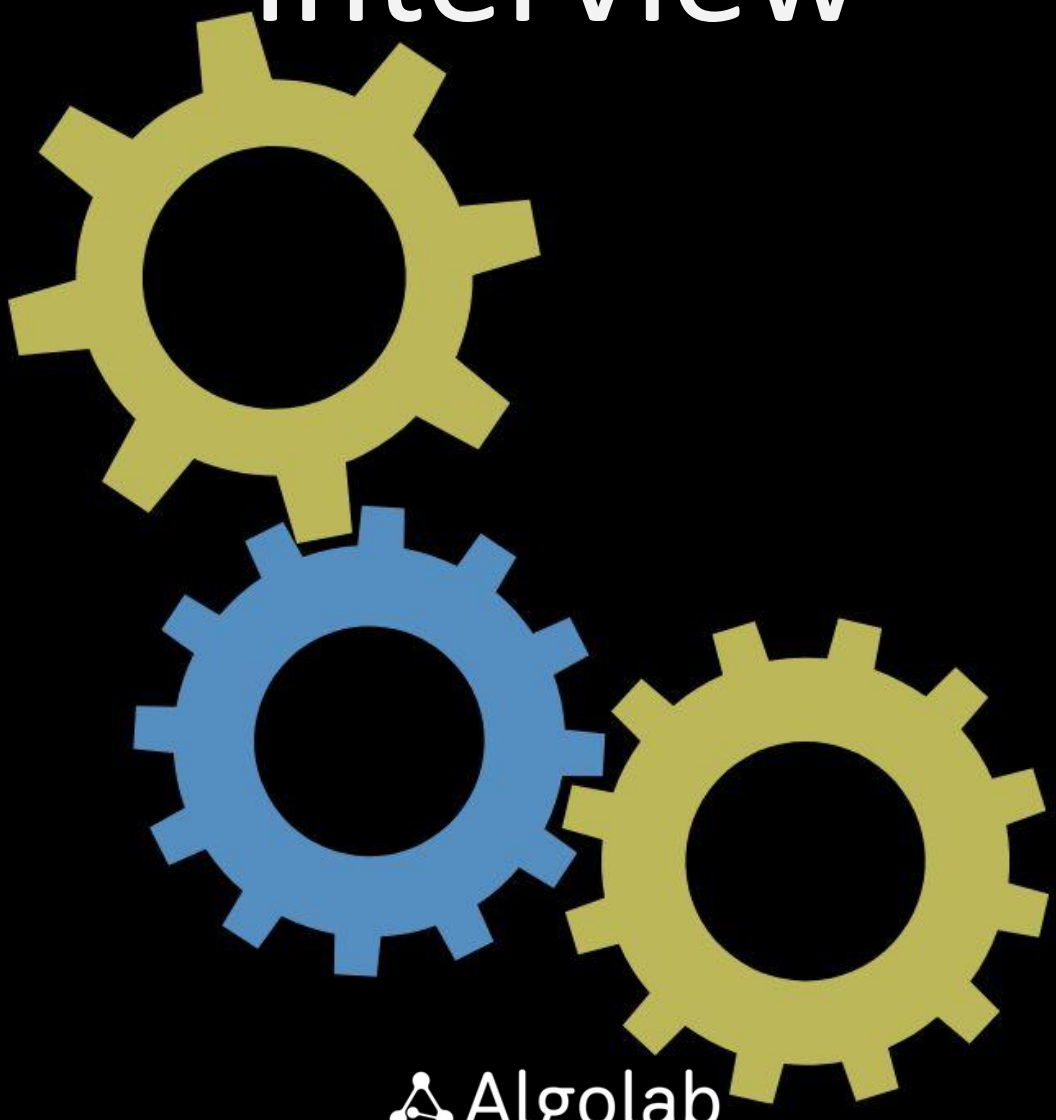
algolab.so

Essentials for System Design Interview



Algolab

CAP theorem (Consistency, Availability, Partition tolerance)

A theoretical result that states that in a distributed system, it is impossible to simultaneously achieve all three of the following guarantees

Consistency

In a distributed system, consistency refers to the property that all nodes see the same data at the same time.

Availability

The ability of a system or service to be operational and accessible to users when needed.

Partition tolerance

The ability of a distributed system to continue operating correctly even when the network is partitioned into multiple isolated subnetworks.

Membership protocol

A protocol that defines how nodes in a distributed system discover and communicate with each other.

Replication protocol

A protocol that defines how data is replicated across multiple nodes in a distributed system.

Epidemic protocol

A protocol used in distributed systems to quickly disseminate information or updates to all nodes in the system.

Gossip protocol

A protocol used in distributed systems to disseminate information or updates to other nodes in an efficient and decentralized manner.

Ring protocol

A protocol used in distributed systems to organize nodes into a logical ring structure, often used for leader election or data dissemination.

Consistency protocol

A protocol that defines how data is kept consistent across multiple nodes in a distributed system.

Fault tolerance protocol

A protocol that defines how a distributed system responds to and recovers from failures or faults.

Quorum-based protocol

A protocol that requires a minimum number of nodes to be available in order for the system to operate correctly.

Merkle tree

A data structure used in distributed systems to efficiently verify the integrity of large datasets.

Distributed hash table (DHT)

A data structure used in distributed systems to store and retrieve data in a decentralized manner.

Consistent hashing

A technique used in distributed systems to evenly distribute keys across a hash table without the need to completely rebuild the table when nodes are added or removed.

Two-phase commit (2PC)

A protocol used in distributed systems to ensure that a transaction either commits on all nodes or rolls back on all nodes.

Paxos

Paxos is the original distributed consensus algorithm, with modified versions used by Chubby and many others. (Zookeeper’s ZAB is similar to Paxos as well.)

Raft

Raft is a distributed consensus algorithm that is used to ensure that a distributed system reaches agreement on the state of its data.

SWIM

SWIM is a distributed gossip protocol for group membership (e.g. for determining members of a caching ring, etc)

Inverted Bloom filters

An inverted Bloom filter is a variant of the Bloom filter that stores the elements that are not present in the filter, rather than the elements that are present. This can be useful for situations where it is more important to minimize false negatives than false positives.

Cuckoo filters

A Cuckoo filter is a probabilistic data structure that is similar to a Bloom filter, but uses a different approach to store and retrieve data. It is designed to be more space-efficient than Bloom filters.

Quotient filters

A quotient filter is a probabilistic data structure that is similar to a Bloom filter, but uses a different approach to store and retrieve data. It is designed to be more space-efficient than Bloom filters, especially for large datasets.

Bloom filters

A standard Bloom filter is the basic version of the Bloom filter that uses a single hash function and a single bit array to store and retrieve data.

Pessimal Congestion Theorem

This theorem states that the maximum possible network congestion in a distributed system is achieved when all nodes send their maximum possible rate of traffic.

Little's Law

This law states that the average number of items in a system (e.g., requests in a queue) is equal to the rate at which items enter the system (e.g., requests per second) multiplied by the average time each item spends in the system (e.g., average response time).

Queueing Theory

This theory is a mathematical study of waiting lines, or queues, and their associated processes. It is commonly used to model and analyze the performance of distributed systems.

Maximum Flow Theorem

This theorem states that the maximum flow through a network from a given source to a given sink is equal to the minimum capacity of any cut (a partition of the nodes) that separates the source from the sink.

Randomized Load Balancing

This principle states that distributing workload randomly among servers can lead to more efficient resource utilization and improved performance in a distributed system.

LP Impossibility Theorem

This theorem, also known as the Fischer-Lynch-Paterson Impossibility Theorem, states that it is impossible to achieve both reliability and consistency in a distributed system with asynchronous communication and no failures.

Amdahl's Law

This law states that the speedup of a parallel system is limited by the fraction of the workload that can be parallelized.

PACELC Theorem

PACELC theorem states that in the case of Network Partition ‘P’ a distributed system can have tradeoffs between Availability ‘A’ and Consistency ‘C’ Else ‘E’ if there is no Network Partition then a distributed system can have tradeoffs between Latency ‘L’ and Consistency ‘C’.

Consistency level

In a distributed system, the consistency level specifies the minimum number of nodes that must respond to a read or write request in order for the request to be considered successful.

Eventual consistency

A consistency model in which all nodes in a distributed system eventually agree on the same value or state, even if there are delays or conflicts in the interim.

Linearizability

A consistency model in which the order of operations is preserved, and all nodes see the same sequence of operations.

Strong consistency

A consistency model in which all nodes in a distributed system see the same data at the same time.

Weak consistency

A consistency model in which there may be delays in data propagation or temporary inconsistencies, but eventually all nodes in the system will agree on the same value or state.

Strong eventual consistency

A consistency model in which all nodes in a distributed system eventually agree on the same value or state, and there are no temporary inconsistencies.

Weak eventful consistency

A consistency model in which there may be temporary inconsistencies, but eventually all nodes in the system will agree on the same value or state.

Serializability

This is a consistency model that ensures that the execution of multiple transactions appears to be serialized, as if they were being executed one at a time in a specific order. This model is weaker than linearizability, but it still ensures that all transactions see a consistent view of the data.

Causal consistency

This is a consistency model that ensures that the order of operations is preserved within the context of a single client. This means that if a client performs an operation A, followed by an operation B, all other clients will see the effects of operation A before the effects of operation B.

Cache

A cache is a temporary storage area that holds frequently accessed data in order to speed up future access to the data. In a distributed cache, the cache is distributed across multiple nodes in a network.

Cache hit

A cache hit occurs when the data being requested is found in the cache. This avoids the need to retrieve the data from the original source, which can be slower.

Cache miss

A cache miss occurs when the data being requested is not found in the cache. This requires the data to be retrieved from the original source.

Cache eviction

Cache eviction is the process of removing items from a cache when the cache becomes full. Different cache eviction algorithms (e.g., least recently used (LRU), least frequently used (LFU)) can be used to determine which items should be removed.

Cache consistency

Cache consistency refers to the accuracy and currency of the data in the cache. In a distributed cache, cache consistency must be maintained across all nodes to ensure that the data is up-to-date and correct.

Cache invalidation

Cache invalidation is the process of marking data in the cache as invalid or out-of-date. This can be done manually or automatically (e.g., based on a time-to-live (TTL) value).

Cache expiration

Cache expiration refers to the automatic removal of data from the cache after a specified period of time. This is used to ensure that the cache does not contain stale or out-of-date data.

Cache sharding

Cache sharding is the process of dividing the cache into smaller pieces, called "shards," and distributing the shards across multiple nodes. This allows the cache to scale horizontally and handle larger workloads.

Cold Cache

A cold cache refers to a cache that has not been recently used and therefore contains stale or no data.

Hot Cache

A hot cache is a cache that has been recently used and contains up-to-date data.

Cache warming

Cache warming is the process of pre-populating a cache with data that is likely to be requested in the future. This can improve the performance of the cache by reducing the number of cache misses.

Cache stampede

A cache stampede is a phenomenon that occurs when multiple processes or threads try to access the same data in a cache simultaneously, resulting in a sudden spike in resource usage and performance degradation. Cache stampedes can be mitigated using techniques such as rate limiting and distributed locks.

Data replication

The process of copying data from one location to another, usually for the purpose of improving availability or fault tolerance.

Distributed ledger

A database that is replicated across multiple nodes and stores a record of all transactions that have occurred within the system.

State machine replication

A technique used in distributed systems to ensure that all replicas of a system are in the same state, even in the presence of failures or network partitions.

Data replication lag

In a distributed system, data replication lag is the time it takes for data to be replicated to all nodes.

Optimistic replication

A replication strategy in which multiple nodes can update data concurrently, and conflicts are resolved as needed.

Replication factor

In a distributed system, the replication factor is the number of copies of data that are stored on different nodes.

Replication lag

In a distributed system, replication lag is the time it takes for data to be replicated to all nodes.

ACID (Atomicity, Consistency, Isolation, Durability)

A set of properties that describe the behavior of a database transaction, ensuring that the transaction is completed in its entirety or not at all, and that the data remains in a consistent state throughout the process.

Byzantine fault tolerance

The ability of a distributed system to operate correctly even when some of its components exhibit arbitrary or malicious behavior.

Distributed system

A system that consists of multiple independent components that work together as a single entity, often over a network.

Fault tolerance

The ability of a system to continue functioning correctly even when one or more of its components fail.

Load balancing

The process of distributing workload evenly across multiple resources in order to improve performance and reliability.

Quorum

In a distributed system, a quorum is a minimum number of nodes that must be available in order for the system to operate correctly.

Scalability

The ability of a system to handle increasing workloads without a decrease in performance.

System architecture

The overall design and structure of a system, including the hardware, software, and communication protocols used.

Consensus

In a distributed system, consensus refers to the process of agreeing on a single value or decision among multiple nodes.

Consistency model

A set of rules that defines how data is accessed and modified in a distributed system, and how data conflicts are resolved.

Fault detection

The process of identifying when a component in a distributed system is not functioning correctly.

Fault injection

The process of intentionally introducing failures or faults into a distributed system in order to test its fault tolerance and error handling capabilities.

Leader election

In a distributed system, leader election is the process of choosing a single node to coordinate the actions of the other nodes.

Network partition

A scenario in which a distributed system becomes divided into multiple isolated subnetworks, often due to a failure or disruption in the network.

Replica

A copy of data that is stored on a different node in a distributed system.

Sharding

The process of dividing a large database into smaller pieces, or shards, and distributing them across multiple nodes in a distributed system.

Hot standby

A configuration in which a secondary node in a distributed system is continuously kept up to date with the primary node, so that it can take over if the primary node fails.

Load balancer

A device or software that distributes workload evenly across multiple resources in a distributed system in order to improve performance and reliability.

Message passing

A communication paradigm in which nodes in a distributed system send messages to each other in order to exchange information or coordinate actions.

Overlay network

A virtual network that is built on top of an existing network, often used to improve the scalability or reliability of distributed systems.

P2P (peer-to-peer)

A type of distributed system in which nodes communicate and exchange data directly with each other, without the need for a central server or authority.

Service discovery

The process of locating and identifying services in a distributed system.

Vector clock

A data structure used in distributed systems to track the relative order of events and detect conflicts or inconsistencies.

Atomicity

In a distributed system, atomicity refers to the property that a transaction either completes in its entirety or not at all, and that the data remains in a consistent state throughout the process.

Centralized system

A system in which all components are controlled by a single entity or authority.

Client-server architecture

A system architecture in which clients request services or data from servers.

Commit

In a distributed system, a commit is the final step in a transaction that indicates that the changes made by the transaction should be made permanent.

Data partitioning

The process of dividing a dataset into smaller pieces and storing them on different nodes in a distributed system.

Durability

In a distributed system, durability refers to the property that once a transaction has been committed, it will not be lost even in the event of a failure or system crash.

Federation

A type of distributed system in which multiple independent systems or organizations collaborate and share resources.

High availability

A system design goal that aims to ensure that a system or service is always operational and accessible to users.

Isolation

In a distributed system, isolation refers to the property that the execution of one transaction does not interfere with the execution of other transactions.

Master-slave architecture

A system architecture in which a single master node controls one or more slave nodes.

Microservices

A software architecture in which a large system is divided into smaller, independently deployable services that communicate with each other through APIs.

Mutual exclusion

In a distributed system, mutual exclusion refers to the property that only one process can access a shared resource at a time.

Rollback

In a distributed system, a rollback is the process of undoing the changes made by a transaction that has not yet been committed.

Symmetric architecture

A system architecture in which all nodes are equal and can communicate with any other node.

Three-tier architecture

A system architecture in which the presentation, application, and data layers are separated and distributed across different nodes.

Asynchronous communication

A communication paradigm in which messages or requests are sent without waiting for a response.

Consistent snapshot

In a distributed system, a consistent snapshot is a copy of the data that is taken at a specific point in time, and is consistent across all nodes in the system.

Decentralized system

A system in which the components are not controlled by a single entity or authority, but rather operate independently and make their own decisions.

Directed acyclic graph (DAG)

A data structure that consists of directed edges and nodes, and does not contain any cycles.

Hybrid architecture

A system architecture that combines elements of centralized and decentralized systems.

Leader-follower architecture

A system architecture in which a single leader node coordinates the actions of one or more follower nodes.

Message passing interface (MPI)

A standard for communicating between nodes in a distributed system, often used for parallel computing.

Multi-master architecture

A system architecture in which multiple nodes can accept writes and updates, and conflicts are resolved through some form of conflict resolution.

Redundancy

The practice of storing multiple copies of data in a distributed system in order to improve reliability and fault tolerance.

Synchronous communication

A communication paradigm in which messages or requests are sent and a response is waited for before continuing.

Active-passive Architecture

A system architecture in which a single active node performs a task or function, while one or more passive nodes stand by as backups.

Consistency window

In a distributed system, the consistency window is the time period during which data is kept consistent across all nodes.

Consistency check

In a distributed system, a consistency check is a process that verifies that all nodes have the same data and are in a consistent state.

Consistency violation

In a distributed system, a consistency violation occurs when the data on one or more nodes is not in a consistent state with the rest of the system.

Data center

A facility that houses a large number of servers and other computing infrastructure.

Data partition

In a distributed system, a data partition is a subset of the data that is stored on a particular node.

Data synchronization

The process of ensuring that data is consistent across all nodes in a distributed system.

Disaster recovery

The process of preparing for and recovering from disasters or disruptions in a distributed system.

Distributed file system

A file system that stores data across multiple nodes in a distributed system.

Distributed lock manager (DLM)

A component in a distributed system that manages locks on shared resources.

Distributed process

A process that executes on multiple nodes in a distributed system.

Distributed transaction

A transaction that spans multiple nodes in a distributed system.

Eventual consistency window

In a distributed system, the eventual consistency window is the time period during which data may be temporarily inconsistent, but will eventually become consistent.

Fault tolerance level

The level of resilience of a distributed system to failures or faults.

Global locking

In a distributed system, global locking is the process of acquiring a lock that spans all nodes in the system.

Hypervisor

A software layer that allows multiple operating systems to run on a single physical host.

Latency

The time it takes for a message or request to be sent and received in a distributed system.

Leaderless architecture

A system architecture in which there is no central authority or leader, and all nodes are equal.

Soft state

In a distributed system, soft state refers to data or state that is not persisted on disk and can be regenerated if lost.

Idempotence

Idempotence describes an operation that always leads to the same outcome, no matter how many times you execute it. If the parameters are the same, an idempotent operation won’t affect the application it calls.

Autoscaling

Autoscaling is a technique that involves automatically adjusting the capacity of a system to handle workloads.

Data partitioning

Data partitioning is the process of dividing the data in a cache into smaller pieces and distributing the pieces across multiple nodes. This allows the cache to scale horizontally and handle larger workloads.

Load balancing

Load balancing is the process of distributing workloads evenly across multiple servers or resources in a distributed system. In a distributed cache, load balancing is used to ensure that the cache remains available and performs optimally.

Distributed locks

Distributed locks are used to synchronize access to shared resources in a distributed system. In a distributed cache, distributed locks can be used to ensure that only one process can update a piece of data at a time.

Secondary index

An index that allows you to query a database table based on an attribute other than the primary key. Secondary indexes can be created at the time the table is created or added to an existing table.

Provisioned throughput

The maximum amount of read and write capacity units that you want to reserve for a DynamoDB table or secondary index. Provisioned throughput determines how many reads and writes per second your DynamoDB table or index can support.

Read capacity unit

One read capacity unit represents one strongly consistent read per second, or two eventually consistent reads per second, for an item up to 4 KB in size.

Write capacity unit

One write capacity unit represents one write per second for an item up to 1 KB in size.

Global secondary index

A secondary index that has a partition key and a sort key that can be different from the primary key of the table. A global secondary index allows you to query the table using an alternate key, in addition to queries on the primary key.

Local secondary index

A secondary index that has the same partition key as the primary key of the table, but a different sort key. A local secondary index allows you to query the table using an alternate sort key, in addition to queries on the primary key.

Point-in-time recovery (PITR)

Point-in-time recovery (PITR) in the context of computers involves systems, often databases, whereby an administrator can restore or recover a set of data or a particular setting from a time in the past.