System Design Interview Cheat Sheet

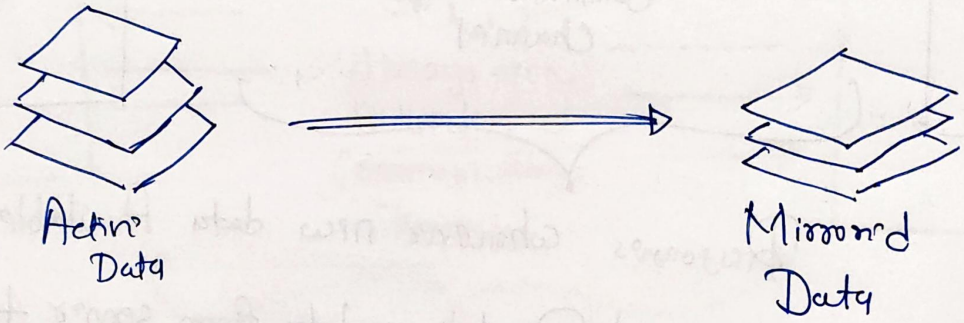
Distributed System Fundamentals

Data Durability and Consistency

The differences and impacts of failure rates of Storage solutions and corruption rates in read-write Processes.

Replication

Backing up data and Replicating Processes at Scale.

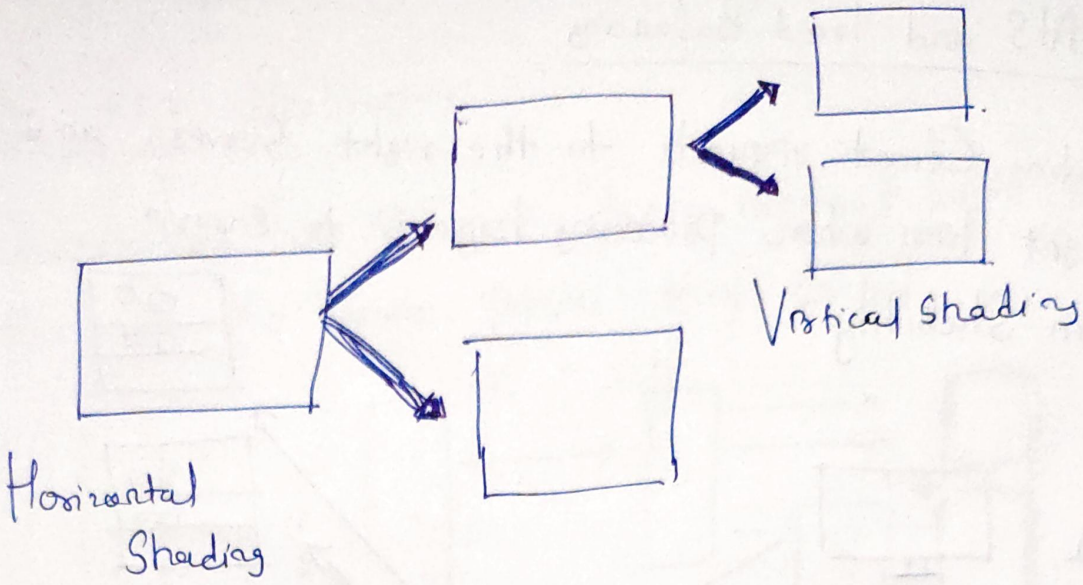


Consensus

Ensuring all nodes are in agreement, which prevent processes from Running and ensures consistency and Replication of data and Processes.

Partitioning

Dividing data across different node within Systems, which Reduces reliance on pure Replication.



Distributed transactions

↳ once consensus is reached, transactions from applications need to be commited across databases with fault checks by each resource involved.

Architecture of Scalable Web Applications

HTTP

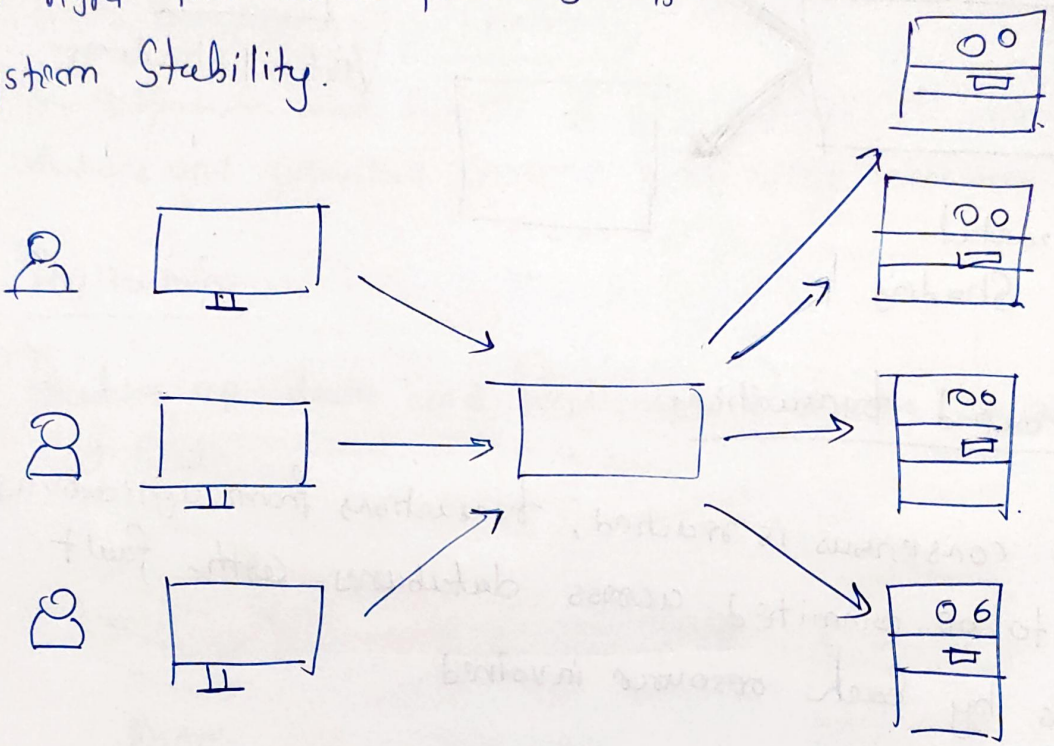
↳ The API on which the Entire internet Runs

REST

The set of design Principles that directly interact with HTTP to Enable System Efficiently and Scalability.

DNS and load Balancing

Routing Client requests to the right Servers and the right tiers when processing happens to Ensure System Stability.



Caching

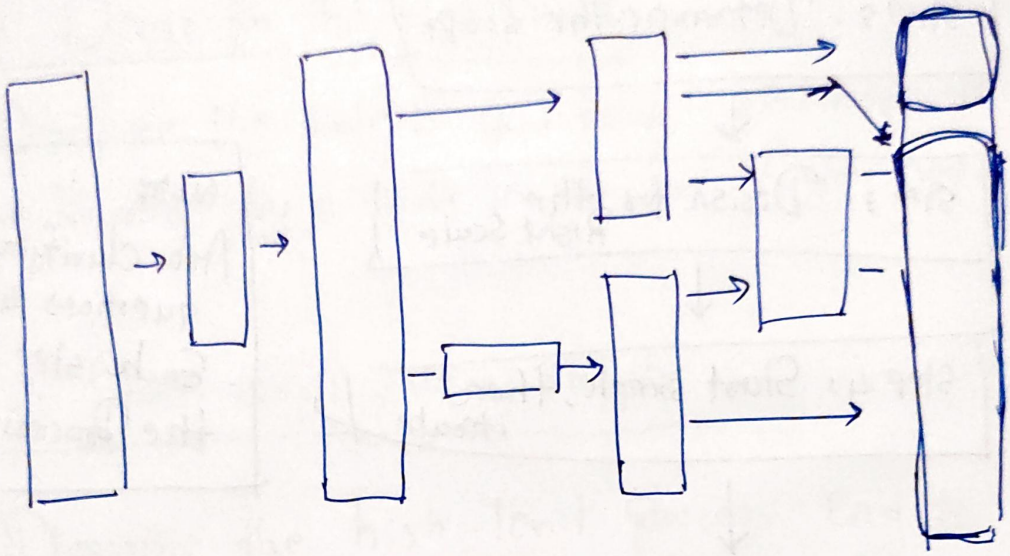
Making tradeoffs and Caching decisions to determine what should be stored in a cache, how to direct traffic to a Cache, and how to Ensure we have the appropriate data in a cache.

N- tier Applications

Understanding how Processes to data streams to allocate for Efficient use of local Resources.

N tier Applications

Understanding how processing tiers interact with each other and the specific process they control



Stream Processing

Applying uniform processes to data streams to allow for efficient use of local Resources.

How to Design Large-Scale Systems

Step 1: Clarify the Goals

↓

Step 2: Determine the Scope

↓

Step 3: Design for the Right Scale

↓

Step 4: Start simple, then iterate

↓

Step 5: Consider Relevant DSA

↓

Step 6: Describe trade-offs

Note:
Ask clarifying questions at Each step of the Process!

Step 1: Clarify the Goals

↳ Make sure you understand the Basic Requirements and ask any clarifying questions.

Step 2: Determine the Scope

↳ Describe the feature set you'll be discussing in the given solution, and define all of the features and their importance to the end goal.

Step 3: Design for the Right Scale

↳ Determine the scale so you know whether the data can be supported by a single machine or if you need to scale.

Step 4: Start simple, then iterate

↳ Describe the high-level process End to End based on your feature set and over all goals. This is a good time to discuss potential bottlenecks.

Step 5: Consider Relevant DSA.

↳ Determine which ~~critical~~ fundamentals data Structures and algorithms will help your System perform Efficiently and appropriately.

Step 6: Describe trade-offs

↳ Describe trade-offs while Expanding Explaining your solution to show you understand large-scale Systems and their Compositions.

Ask clarifying Questions at Each Step of the Process!