

CheatSheet: System Design For Job Interview

INTERVIEW

- PDF Link: [cheatsheet-systemdesign-A4.pdf](#), Category: interview
- Blog URL: <https://cheatsheet.dennyzhang.com/cheatsheet-systemdesign-A4>
- Related posts: CheatSheet: Leetcode For Code Interview, CheatSheet: Well-Known Papers For IT Industry, #denny-cheatsheets

File me Issues or star this repo.

1.1 Reference

Name	Summary
Papers	CheatSheet: Well-Known Papers For IT Industry, Github: papers-we-love
Github	Github: system-design-primer
Cheatsheet	CheatSheet: Leetcode For Code Interview, CheatSheet: Common Code Problems & Follow-ups
Cheatsheet	CheatSheet: System Design For Job Interview, CheatSheet: SRE/DevOps/Sysadmin
Cheatsheet	CheatSheet: Behavior Questions For Coder Interview
Coding	Code problems for #oodeign
YouTube	YouTube: Intro to Architecture and Systems Design Interviews
YouTube	YouTube Channel: Success in Tech, YouTube: Scalability Harvard Web Development
YouTube	YouTube: System Design Interview

1.2 Process Of System Design

Num	Name	Summary
1	Outline use cases: List major and focus on some	Show good sense. The questions you asked define your level
2	Estimate scale: Data + Traffic	Back-of-the-envelope estimation
3	Defining data model	It helps to clarify how data will flow among different components
4	Abstract design	Sketch main components, explain workflow, avoid too deep for details
5	Detailed design + discussion with interviewers	Explain trade-off of your proposal + on-demand deep dive
6	Identify and resolve Bottlenecks	Key challenges + Trade-Offs . Usuaully no optimal solution(s)
7	Scale your design	Availability, Resiliency, Scalability, Security, Serviceability, etc

1.3 Top 20 Design Problems For Technical Modules

Num	Name
1	Design a distributed counter
2	Delayed task scheduling
3	Design a thread-safe Hashmap
4	Design An API Rate Limiter
5	Delayed queue
6	Design a distributed UUID generator
7	Design a distributed Hashmap
8	Design data sync for a distributed system
9	Design A big file transfer feature
10	Design a distributed transactions
11	Design: A Parking Lot Service
12	Design a distributed transaction
13	Design: A URL Redirecting Feature
14	Top URL hits
15	Unique url hits
16	Design a load balancer
17	Design a client-server API to build a rich document editor
18	Design online/offline status system
19	Design a circuit breaker
20	Design a service auto-discovery feature
21	Design a secrets management system
22	Design a online contest system like leetcode.com

1.4 Top 20 Design Problems For A Product

Num	Name
1	Design: TinyURL - A URL Shortener Service
2	Design Twitter News Feed
3	Design K/V DB
4	Design: Leaderboard
5	Design: Flight booking service
6	Design: Uber Backend
7	Design an API gateway
8	Design: An Elevator Service
9	Design web crawler
10	Design amazon shopping cart
11	Design: Google Suggestion Service
12	Design a payment processor
13	Design google doc
14	Design gmail
15	Design instagram, a photo sharing app
16	Design Yelp, a location-based system
17	Design Pastebin.com
18	Design amazon book recommendation system
19	Google autocomplete
20	Design Google PageRank
21	Design messaging/notification system
22	Design search post system
23	Design memcache/redis
24	Design typeahead
25	Design Google AdSense fraud detection
26	Design a voice conference system
27	Design slack

1.5 Top 30 Concepts For Feature/System Design

Num	Name	Summary
1	Caching	Stores data so that future requests of data retrieval can be faster Provides an asynchronous communications protocol, Break up a big data volume into many smaller parts Create indexes on multiple columns to speed up table look up Duplicate data to increase service availability A distributed database system can only have 2 of the 3 Relational databases and non-relational databases
2	Message Queue	
3	Data Partition & Sharding	
4	DB Indexing	
5	DB replication	
6	CAP: Consistency/Availability/Partition	
7	DB: SQL & NoSQL	
8	Concurrency & Communication	weak consistency, eventual consistency, strong consistency Quorum, vector lock, reconcile on read/write, CRDTs
9	Pessimistic And Optimistic Locking	
10	Consistency Module	
11	Conflict resolution	
12	B+ Tree	
13	Networking: HTTP	API Rate limit, Circuit breaker, bulkhead, throttling
14	Pull vs Push model	
15	Garbage Collection	
16	Memory Management	
17	Heartbeats	
18	Self Protection	Vertical scaling and Horizontal scaling
19	Filesystem	
20	API: gRPC vs REST	
21	Load balancer	
22	Scale up vs Scale out	
23	API Design	Weak consistency, Eventual consistency, Strong consistency Fail-over vs Replication Edge caching
24	Session management	
25	Networking: TCP vs UDP	
26	Consistency patterns	
27	Availability patterns	
28	CDN - Content Delivery Network	
29	Monitoring	
30	Security	
31	Networking: DNS	
32	Linux signals	

1.6 Top 15 Advanced Data Structure & Algorithms

Num	Name	Summary
1	Consistent Hash	A space-efficient query returns either "possibly in set" or "definitely not in set"
3	Bloom filter	
4	CRDTs (Conflict-Free Replicated Data Types)	
5	SSTable (Sorted Strings Table)	Propagate cluster status
6	LSM (Log Structured Merge Trees)	
7	Gossip	
8	Two-phase commit/Three-phase commit	
10	Vector Clocks/Version Vectors	
11	Paxos and raft protocol	
12	Merkle Tree	

<https://raw.githubusercontent.com/dennyzhang/cheatsheet.dennyzhang.com/master/cheatsheet-featuredesign-A4/dynamo-summary.png>

1.7 Explain workflow: What happens when XXX?

Num	Name	Summary
1	When happens when I search in google?	
2	How loadbalancer works	
3	Explain three phase commit model	
4	Explain HTTP return code	
5	Explain Mysql DB replication model	
6	Explain gossip protocol	
7	Explain CAP	
8	Explain Hadoop file system	
9	[Linux] Explain OS booting process	

1.8 Explain tools: how XXX supports XXX?

Num	Name	Summary
1	How JDK implement hashmap?	
2	Explain java garbage collection model	
3	Explain raft/etcd	
4	How OS supports XXX?	

1.9 Cloud Design Principles

Num	Name	Summary
1	Fail fast	
2	Design for failure	
3	Immutable infrastructure	
4	Cats vs Cattle	Avoid snowflake servers
5	Auto healing	
6	Async programming	
7	GitOps operational model	
8	Event-Driven Architectures	

1.10 Cloud Design Patterns

Num	Name	Summary
1	Ambassador pattern	Create helper service to send network requests, besides the main service
2	Cache-Aside pattern	Load data on demand into a cache from a data store
3	Circuit Breaker pattern	If a request takes too many resource, abort it
4	Bulkhead pattern	Isolate elements into pools, so that one fire won't burn all
5	Gateway Aggregation pattern	Aggregate multiple individual requests into a single request
6	Priority Queue pattern	Support different SLAs for different individual clients
7	Strangler pattern	Incrementally migrate a legacy system piece by piece

1.11 Engineering Of Well-Known Products

Name	Summary
Google	Link: Google Architecture
Facebook	Link: Facebook Live Streams
Twitter	Link: Twitter Image Service, YouTube: Timelines at Scale
Uber	Link: Lessons Learned From Scaling Uber
Tumblr	Link: Tumblr Architecture
StackOverflow	Link: Stack Overflow Architecture

1.12 Grow Design Expertise In Daily Work

Num	Name	Summary
1	Keep the curiosity	Thinking about interesting/weird questions helps
2	Deep dive into your daily work	Unify and normalize problems from daily work
3	Learn the work of your colleagues	Indirect working experience also help
4	Popular products under the hood	Once you notice an interesting feature, think about how it's supported?
5	Read engineering blogs	Especially for big companies
6	Tools under the hood	Common tools/frameworks
7	Try tools	Use cases; Alternatives; Pros and Cons
8	Read papers	Best practices in papers
9	Try new things	Gain hands-on experience; evaluate alternatives
10	Datastore & OS	Learn how databases and operating systems work
11	Language implementation	Deep dive into one programming language. Java, Python, Golang, etc

1.13 Engineering Blogs/Websites

Name	Summary
Compnay Tech Blog	Website: Facebook Engineering, Website: Google Developers
Compnay Tech Blog	Medium: Netflix Blog, Medium: Airbnb Engineering & Data Science
Compnay Tech Blog	Shopify Engineering, Github Engineering
Website	Website: hiredintech - System Design
Website	Website: interviewing.io, Website: interviewbit.com
Reference	Link: Preparing for your Software Engineering Interview at Facebook
Reference	Link: The System Design Process
Individual Tech Blog	Blog: All Things Distributed - Amazon CTO, Blog: highscalability

1.14 Typical Trade-Off

Num	Name	Summary
1	Performance vs Scalability	
2	Latency vs Throughput	
3	Availability vs Consistency	Brewer's CAP theorem

1.15 Misc

Num	Name	Summary
1	How to store 2TB data into 3 disks of 1TB. And be tolerant for one disk failure	A, B, C. And $C = A \oplus B$
2	Find out the difference between two files. Majority of these two are the same	#lcs - Longest Common Subsequence
3	How to support feature of "diff 1.txt 2. txt"	
4	Avoid double payment in a distributed payment system	link

1.16 More Resources

License: Code is licensed under MIT License.

<https://github.com/binhnguyennus/awesome-scalability>

<https://github.com/donnemartin/system-design-primer>

<https://github.com/checkcheckzz/system-design-interview>

<https://github.com/binhnguyennus/awesome-scalability>

<https://docs.microsoft.com/en-us/azure/architecture/patterns/>

<https://github.com/sdmg15/Best-websites-a-programmer-should-visit>