

System Design Interview Prep Master Doc

All system design youtube channels

1. <https://www.youtube.com/user/dimakorolev/videos>
2. <https://www.youtube.com/c/ByteByByte/videos>
3. https://www.youtube.com/channel/UC_n-A84J0UcU5uq4sEh2CnQ
4. <https://www.youtube.com/c/DefogTech/videos>
5. <https://www.youtube.com/c/HusseinNasser-software-engineering/videos>
6. <https://www.youtube.com/c/SystemDesignInterview/videos>
7. <https://www.youtube.com/c/sudoCODE/videos>
8. <https://www.youtube.com/c/codeKarle/videos>
9. <https://www.youtube.com/c/CMUDatabaseGroup/videos>
10. <https://www.youtube.com/c/interviewingio/videos>
11. <https://www.youtube.com/c/GauravSensei/videos>
12. <https://www.youtube.com/c/EngineeringwithUtsav/videos>
13. <https://www.youtube.com/channel/UCIB4KPy5LkJ1t3SgYVtMOQ/videos>
14. <https://www.youtube.com/c/ExponentTV/videos>
15. https://www.youtube.com/watch?v=cQP8WApzIQQ&list=PLrw6a1wE39_tb2fErI4-WkMbsvGQk9_UB
16. <https://www.youtube.com/c/TheInterviewSage/videos>
17. <https://www.youtube.com/c/ThinkSoftware/videos>
18. <https://www.youtube.com/c/SuccessinTech/videos>
19. <https://www.youtube.com/c/TechDummiesNarendraL/videos>
20. <https://www.youtube.com/c/OktaDev/videos>

Important Books :

1. [Amazon.com: System Design Interview – An Insider's Guide eBook: Xu, Alex: Kindle Store](#)
2. [Amazon.com: Operating Systems: Three Easy Pieces eBook: Arpaci-Dusseau, Remzi, Arpaci-Dusseau, Andrea: Kindle Store](#)
3. [Web Scalability for Startup Engineers: Ejsmont, Artur: 9780071843652: Amazon.com: Books](#)
4. [Understanding Distributed Systems: What every developer should know about large distributed applications: Vitillo, Roberto: 9781838430207: Amazon.com: Books](#)
5. [Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems: Kleppmann, Martin: 9781449373320: Amazon.com: Books](#)
6. [Kafka: The Definitive Guide: Real-Time Data and Stream Processing at Scale: 9781491936160: Computer Science Books @ Amazon.com](#)
7. [Buy Distributed Algorithms – An Intuitive Approach 2e \(The MIT Press\) Book Online at Low Prices in India | Distributed Algorithms – An Intuitive Approach 2e \(The MIT Press\) Reviews & Ratings - Amazon.in](#)
8. [Elasticsearch: The Definitive Guide: A Distributed Real-Time Search and Analytics Engine eBook: Gormley, Clinton, Tong, Zachary: Amazon.in: Kindle Store](#)
9. [Buy Cassandra – The Definitive Guide, 3e: Distributed Data at Web Scale Book Online at Low Prices in India | Cassandra – The Definitive Guide, 3e: Distributed Data at Web Scale Reviews & Ratings - Amazon.in](#)

Imp keywords

1. Write ahead logging
2. SSL passthrough / termination on load balancers
3. Clock vectors
4. Hinted handoff
5. Bloom filters
6. Gossip protocols
7. Merkle trees
8. Two phase commit
9. Two phase locking
10. Consistent hashing
11. Data replication
12. Total order broadcast
13. Isolation levels (read uncommitted, read committed, repeatable read, serializable)
14. Quad trees (GeoHashin)
15. Inverse indexing - Google search/any search indexing
16. Gaming ranking - rank players based on score and faster.
17. Word search in trie - auto suggestions system design
18. Sort 5gb data in 1gb memory - merge sort
19. Paxos algo
20. Merkle Tree
21. Backpressure
22. Circuit breaker
23. Raft
24. Service discovery
25. Saga
26. Hyperloglog

Important algorithms

<https://github.com/resumejob/system-design-algorithms>

- Frugal Streaming
- Geohash / S2 Geometry
- Leaky bucket / Token bucket
- Loosy Counting
- Operational transformation
- Quadtree / Rtree
- Ray casting
- Reverse index
- Rsync algorithm
- Trie algorithm
- Consistent Hashing
- Count-Min Sketch
- Bloom Filters
- HyperLogLog
- Skip Lists
- LRU
- B tree

- Hierarchical Timing Wheels <https://blog.acolyer.org/2015/11/23/hashed-and-hierarchical-timing-wheels/>
- Merkle trees
- Fenwick Tree
 - <http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=2FDD6F53D3DC3BB91FA42E7277B6765B?doi=10.1.1.14.8917&rep=rep1&type=pdf>
- AMS (Alon Matias Szegedy) algorithm

Cloud design patterns

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

Cloud arch pattern

Ambassador	Create helper services that send network requests on behalf of a consumer service or application.	Design and Implementation , Operational Excellence
Anti-Corruption Layer	Implement a façade or adapter layer between a modern application and a legacy system.	Design and Implementation , Operational Excellence
Asynchronous Request-Reply	Decouple backend processing from a frontend host, where backend processing needs to be asynchronous, but the frontend still needs a clear response.	Messaging
Backends for Frontends	Create separate backend services to be consumed by specific frontend applications or interfaces.	Design and Implementation

<u>Bulkhead</u>	Isolate elements of an application into pools so that if one fails, the others will continue to function.	<u>Reliability</u>
<u>Cache-Aside</u>	Load data on demand into a cache from a data store	<u>Data Management,</u> <u>Performance Efficiency</u>
<u>Choreography</u>	Let each service decide when and how a business operation is processed, instead of depending on a central orchestrator.	<u>Messaging,</u> <u>Performance Efficiency</u>
<u>Circuit Breaker</u>	Handle faults that might take a variable amount of time to fix when connecting to a remote service or resource.	<u>Reliability</u>
<u>Claim Check</u>	Split a large message into a claim check and a payload to avoid overwhelming a message bus.	<u>Messaging</u>
<u>Compensating Transaction</u>	Undo the work performed by a series of steps, which together define an eventually consistent operation.	<u>Reliability</u>
<u>Competing Consumers</u>	Enable multiple concurrent consumers to process messages received on the same messaging channel.	<u>Messaging</u>
<u>Compute Resource Consolidation</u>	Consolidate multiple tasks or operations into a single computational unit	<u>Design and Implementation</u>
<u>CQRS</u>	Segregate operations that read data from operations that update data by using separate interfaces.	<u>Data Management,</u> <u>Design and Implementation,</u> <u>Performance Efficiency</u>
<u>Deployment Stamps</u>	Deploy multiple independent copies of application components, including data stores.	<u>Reliability,</u> <u>Performance Efficiency</u>
<u>Event Sourcing</u>	Use an append-only store to record the full series of events that describe actions taken on data in a domain.	<u>Data Management,</u> <u>Performance Efficiency</u>
<u>External Configuration Store</u>	Move configuration information out of the application deployment package to a centralized location.	<u>Design and Implementation,</u>

		Operational Excellence
Federated Identity	Delegate authentication to an external identity provider.	Security
Gatekeeper	Protect applications and services by using a dedicated host instance that acts as a broker between clients and the application or service, validates and sanitizes requests, and passes requests and data between them.	Security
Gateway Aggregation	Use a gateway to aggregate multiple individual requests into a single request.	Design and Implementation, Operational Excellence
Gateway Offloading	Offload shared or specialized service functionality to a gateway proxy.	Design and Implementation, Operational Excellence
Gateway Routing	Route requests to multiple services using a single endpoint.	Design and Implementation, Operational Excellence
Geodes	Deploy backend services into a set of geographical nodes, each of which can service any client request in any region.	Reliability, Operational Excellence
Health Endpoint Monitoring	Implement functional checks in an application that external tools can access through exposed endpoints at regular intervals.	Reliability, Operational Excellence
Index Table	Create indexes over the fields in data stores that are frequently referenced by queries.	Data Management, Performance Efficiency
Leader Election	Coordinate the actions performed by a collection of collaborating task instances in a distributed application by electing one instance as the leader that assumes responsibility for managing the other instances.	Design and Implementation, Reliability

Materialized View	Generate prepopulated views over the data in one or more data stores when the data isn't ideally formatted for required query operations.	Data Management , Operational Excellence
Pipes and Filters	Break down a task that performs complex processing into a series of separate elements that can be reused.	Design and Implementation , Messaging
Priority Queue	Prioritize requests sent to services so that requests with a higher priority are received and processed more quickly than those with a lower priority.	Messaging , Performance Efficiency
Publisher/Subscriber	Enable an application to announce events to multiple interested consumers asynchronously, without coupling the senders to the receivers.	Messaging
Queue-Based Load Leveling	Use a queue that acts as a buffer between a task and a service that it invokes in order to smooth intermittent heavy loads.	Reliability , Messaging , Resiliency , Performance Efficiency
Retry	Enable an application to handle anticipated, temporary failures when it tries to connect to a service or network resource by transparently retrying an operation that's previously failed.	Reliability
Scheduler Agent Supervisor	Coordinate a set of actions across a distributed set of services and other remote resources.	Messaging , Reliability
Sequential Convoy	Process a set of related messages in a defined order, without blocking processing of other groups of messages.	Messaging
Sharding	Divide a data store into a set of horizontal partitions or shards.	Data Management , Performance Efficiency
Sidecar	Deploy components of an application into a separate process or container to provide isolation and encapsulation.	Design and Implementation , Operational Excellence

Static Content Hosting	Deploy static content to a cloud-based storage service that can deliver them directly to the client.	Design and Implementation , Data Management , Performance Efficiency
Strangler Fig	Incrementally migrate a legacy system by gradually replacing specific pieces of functionality with new applications and services.	Design and Implementation , Operational Excellence
Throttling	Control the consumption of resources used by an instance of an application, an individual tenant, or an entire service.	Reliability , Performance Efficiency
Valet Key	Use a token or key that provides clients with restricted direct access to a specific resource or service.	Data Management , Security

Data Management patterns

- 06/23/2017
- +3

Data management is the key element of cloud applications, and influences most of the quality attributes. Data is typically hosted in different locations and across multiple servers for reasons such as performance, scalability or availability, and this can present a range of challenges. For example, data consistency must be maintained, and data will typically need to be synchronized across different locations.

Additionally data should be protected at rest, in transit, and via authorized access mechanisms to maintain security assurances of confidentiality, integrity, and availability. Refer to the Azure Security Benchmark [Data Protection Control](#) for more information.

Pattern	Summary
Cache-Aside	Load data on demand into a cache from a data store
CQRS	Segregate operations that read data from operations that update data by using separate interfaces.
Event Sourcing	Use an append-only store to record the full series of events that describe actions taken on data in a domain.

Index Table	Create indexes over the fields in data stores that are frequently referenced by queries.
Materialized View	Generate prepopulated views over the data in one or more data stores when the data isn't ideally formatted for required query operations.
Sharding	Divide a data store into a set of horizontal partitions or shards.
Static Content Hosting	Deploy static content to a cloud-based storage service that can deliver them directly to the client.
Valet Key	Use a token or key that provides clients with restricted direct access to a specific resource or service.

DATA MANAGEMENT PATTERNS

Design and Implementation patterns

- 06/23/2017
- 2 minutes to read

Good design encompasses factors such as consistency and coherence in component design and deployment, maintainability to simplify administration and development, and reusability to allow components and subsystems to be used in other applications and in other scenarios. Decisions made during the design and implementation phase have a huge impact on the quality and the total cost of ownership of cloud hosted applications and services.

Pattern	Summary
Ambassador	Create helper services that send network requests on behalf of a consumer service or application.
Anti-Corruption Layer	Implement a façade or adapter layer between a modern application and a legacy system.
Backends for Frontends	Create separate backend services to be consumed by specific frontend applications or interfaces.
CQRS	Segregate operations that read data from operations that update data by using separate interfaces.
Compute Resource Consolidation	Consolidate multiple tasks or operations into a single computational unit
External Configuration Store	Move configuration information out of the application deployment package to a centralized location.
Gateway Aggregation	Use a gateway to aggregate multiple individual requests into a single request.

Gateway Offloading	Offload shared or specialized service functionality to a gateway proxy.
Gateway Routing	Route requests to multiple services using a single endpoint.
Leader Election	Coordinate the actions performed by a collection of collaborating task instances in a distributed application by electing one instance as the leader that assumes responsibility for managing the other instances.
Pipes and Filters	Break down a task that performs complex processing into a series of separate elements that can be reused.
Sidecar	Deploy components of an application into a separate process or container to provide isolation and encapsulation.
Static Content Hosting	Deploy static content to a cloud-based storage service that can deliver them directly to the client.
Strangler Fig	Incrementally migrate a legacy system by gradually replacing specific pieces of functionality with new applications and services.

DESIGN AND IMPLEMENTATION PATTERNS

Messaging patterns

+4

The distributed nature of cloud applications requires a messaging infrastructure that connects the components and services, ideally in a loosely coupled manner in order to maximize scalability. Asynchronous messaging is widely used, and provides many benefits, but also brings challenges such as the ordering of messages, poison message management, idempotency, and more.

Pattern	Summary
Asynchronous Request-Reply	Decouple backend processing from a frontend host, where backend processing needs to be asynchronous, but the frontend still needs a clear response.
Claim Check	Split a large message into a claim check and a payload to avoid overwhelming a message bus.
Choreography	Have each component of the system participate in the decision-making process about the workflow of a business transaction, instead of relying on a central point of control.
Competing Consumers	Enable multiple concurrent consumers to process messages received on the same messaging channel.
Pipes and Filters	Break down a task that performs complex processing into a series of separate elements that can be reused.
Priority Queue	Prioritize requests sent to services so that requests with a higher priority are received and processed more quickly than those with a lower priority.

Publisher-Subscriber	Enable an application to announce events to multiple interested consumers asynchronously, without coupling the senders to the receivers.
Queue-Based Load Leveling	Use a queue that acts as a buffer between a task and a service that it invokes in order to smooth intermittent heavy loads.
Scheduler Agent Supervisor	Coordinate a set of actions across a distributed set of services and other remote resources.
Sequential Convoy	Process a set of related messages in a defined order, without blocking processing of other groups of messages.

MESSAGING PATTERNS

Book <https://learning.oreilly.com/library/view/cloud-architecture-patterns/9781449357979/>

LEETCODE all system design problems

1. Web Crawler']
<https://leetcode.com/discuss/interview-question/system-design/124657/Facebook-or-System-Design-or-A-web-crawler-that-will-crawl-Wikipedia>
2. Detect web crawler <https://leetcode.com/discuss/interview-question/system-design/548816/Amazon-or-System-Design-or-Web-Crawler-Detector>
3. Yelp
4. Distributed file system
5. URL shortening and Pastebin <https://leetcode.com/discuss/interview-question/system-design/124804/Design-Pastebin>
<https://leetcode.com/discuss/interview-question/system-design/124658/Design-URL-Shortening-service-like-TinyURL>
6. Instagram
<https://leetcode.com/discuss/interview-question/system-design/124802/Design-Instagram>
<https://leetcode.com/discuss/interview-question/system-design/586749/design-Instagram>
<https://leetcode.com/discuss/interview-question/system-design/719253/Design-Facebook-%3A-System-Design-Interview>
7. Dropbox.
8. Twitter
<https://leetcode.com/discuss/interview-question/system-design/124689/Design-twitter>
9. Redis <https://leetcode.com/discuss/interview-question/system-design/125751/Design-a-distributed-cache-system>
10. Youtube or Netflix <https://leetcode.com/discuss/interview-question/system-design/733520/Design-YouTube-Very-detailed-design-with-diagrams>
<https://leetcode.com/discuss/interview-question/system-design/144287/Design-Recommendation-System-for-Amazon-Videos>
<https://leetcode.com/discuss/interview-question/system-design/600861/System-Design-Youtube-add-click-counts>
<https://leetcode.com/discuss/interview-question/system-design/557250/Design-a-video-streaming-service-to-support-playback-video-from-different-devices>

<https://leetcode.com/discuss/interview-question/system-design/496042/Design-video-sharing-platform-like-Youtube>
<https://leetcode.com/discuss/interview-question/system-design/158698/Distributed-database%3A-Netflix>
<https://leetcode.com/discuss/interview-question/system-design/124565/Design-Netflix-recommendation-engine>
<https://leetcode.com/discuss/interview-question/system-design/150607/Design-youtube>

11. Ticketmaster

<https://leetcode.com/discuss/interview-question/system-design/124803/Design-BookMyShow>
<https://leetcode.com/discuss/interview-question/system-design/315763/System-Design-or-Seat-reservation-application-like-Ticket-Master-or-BookMyShow>

12. Facebook Messenger or WhatsApp <https://leetcode.com/discuss/interview-question/system-design/585930/Amazon-or-System-Design-or-Design-a-Chat-Service>
<https://leetcode.com/discuss/interview-question/system-design/220073/How-would-you-design-WhatsApp>
<https://leetcode.com/discuss/interview-question/system-design/124613/Amazon-or-System-Design-or-A-scalable-chat-application-on-phone-browsing>

13. Typeahead suggestions.

14. Twitter search.

15. Newsfeed ranking

<https://leetcode.com/discuss/interview-question/system-design/349627/How-do-you-design-a-meta-data-for-a-news-feed>
[https://leetcode.com/discuss/interview-question/system-design/153871/Design-a-News-Feed-system-\(like-Facebook-LinkedIn-etc.\)](https://leetcode.com/discuss/interview-question/system-design/153871/Design-a-News-Feed-system-(like-Facebook-LinkedIn-etc.))

16. Web search.

17. LinkedIn "you may know ...", <https://leetcode.com/discuss/interview-question/system-design/1036762/Google-Onsite-System-Design-How-to-do-it>
<https://leetcode.com/discuss/interview-question/system-design/153941/Design-the-%22People-You-May-Know%22-feature-on-LinkedIn-or-Facebook>.

18. Uber / Luxe (anti-uber)?

19. Freight / delivery orchestration? (edited)

20. Botnet/decentralized web crawler/torrent

<https://leetcode.com/discuss/interview-question/system-design/594844/System-design-question-Help-needed>
<https://leetcode.com/discuss/interview-question/system-design/464997/Design-a-P2P-file-sharing-application-like-BitTorrent>

21. Coupon redeeming system

<https://leetcode.com/discuss/interview-question/system-design/353302/Design-a-couponvoucher-management-system-or-DellEMC>
<https://leetcode.com/discuss/interview-question/system-design/459593/Facebook-or-System-Design-or-E-commerce-Apply-discount-on-every-nth-order>

22. Message queue kafka, service bus

<https://leetcode.com/discuss/interview-question/system-design/124761/Deciding-which-queue-to-send-a-post-to>
<https://leetcode.com/discuss/interview-question/system-design/206134/Amazon-or-System-Design-or-Design-a-Distributed-Message-queue>
<https://leetcode.com/discuss/interview-question/system-design/734303/Microsoftor-Design-an-Enterprise-Service-Bus>

23. Rate limiter <https://leetcode.com/discuss/interview-question/system-design/637402/Design-a-efficient-client-side-rate-limit-handler>

<https://leetcode.com/discuss/interview-question/system-design/124558/Uber-or-Rate-Limiter>

24. Design leetcode - asked in amazon and fb. <https://leetcode.com/discuss/interview-question/system-design/649021/Design-Leetcode>

25. <https://leetcode.com/discuss/interview-question/system-design/409736/Facebook-or-System-Design-or-Hacker-Rank-LeetCode-Contest-Leadership-Board-System>

<https://leetcode.com/discuss/interview-question/system-design/308452/System-Design-or-Programming-contest-platform-like-LeetCode>

26. Large log data collection and processing system

<https://leetcode.com/discuss/interview-question/system-design/124603/Amazon-or-Phone-screen-or-How-to-handle-large-log-data>

<https://leetcode.com/discuss/interview-question/system-design/128037/How-would-you-parse-a-huge-log-file>

<https://leetcode.com/discuss/interview-question/system-design/189030/Design-a-system-which-can-report-frequently-occurring-exceptions-on-a-dashboard>

<https://leetcode.com/discuss/interview-question/system-design/196142/Copy-coredump-files-from-millions-of-system-to-single-Storage-server-like-S3>

<https://leetcode.com/discuss/interview-question/system-design/431023/Google-or-Onsite-or-Get-all-logs-between-times>

<https://leetcode.com/discuss/interview-question/system-design/440546/Facebook-or-System-Design-Onsite-or-Compute-Percentile-Metrics-Over-Time-Series>

<https://leetcode.com/discuss/interview-question/system-design/124603/Amazon-or-Phone-screen-or-How-to-handle-large-log-data>

<https://leetcode.com/discuss/interview-question/system-design/1133962/Service-which-will-download-data-from-multiple-sources-and-ingests-it-in-the-system>

<https://leetcode.com/discuss/interview-question/system-design/942087/System-Design%3A-Design-a-system-to-process-data-in-different-formats-from-different-sources>

<https://leetcode.com/discuss/interview-question/system-design/852238/Need-help-with-System-Design-problem-asked-in-a-real-interview>

<https://leetcode.com/discuss/interview-question/system-design/820877/Bloomberg-System-Design>

<https://leetcode.com/discuss/interview-question/system-design/778868/Facebook-or-or-Onsite-or-or-System-Design-Aggregation-click-events>

<https://leetcode.com/discuss/interview-question/system-design/725364/System-Design-or-IOT-sensor-data-aggregator>

[https://leetcode.com/discuss/interview-question/system-design/202946/Design-a-system-to-aggregate-metrics-from-large-cluster\(800%2B\)-of-web-servers](https://leetcode.com/discuss/interview-question/system-design/202946/Design-a-system-to-aggregate-metrics-from-large-cluster(800%2B)-of-web-servers)

27. Realtime stock price monitoring system/ live score update cricbuzz, realtime gaming score

<https://leetcode.com/discuss/interview-question/system-design/625918/Amazon-or-System-Design-or-Design-a-real-time-gaming-ranking-system>

<https://leetcode.com/discuss/interview-question/system-design/431712/Bloomberg-or-Design-a-system-to-give-prices-of-a-stock>

28. Stock trading system <https://medium.com/@narengowda/stock-exchange-system-design-answered-ad4be1345851>

<https://leetcode.com/discuss/interview-question/system-design/820877/Bloomberg-System-Design>

<https://leetcode.com/discuss/interview-question/system-design/124794/Design-a-Multicurrency-trading-system>

<https://leetcode.com/discuss/interview-question/system-design/490034/FAANG-or-Onsite-or-Intern-or-System-Design-Stock>

29. Kill switch for stopping stock trading <https://leetcode.com/discuss/interview-question/system-design/124553/Kill-Switch>
30. Design network fail over
<https://leetcode.com/discuss/interview-question/system-design/124598/Design-network-fail-over>
31. Design AB testing framework <https://leetcode.com/discuss/interview-question/system-design/124595/AB-Testing>
<https://leetcode.com/discuss/interview-question/system-design/228661/Design-a-Data-Experimentation-platform>
32. Design parking lot system <https://leetcode.com/discuss/interview-question/system-design/124576/Design-a-parking-lot-system>.
<https://leetcode.com/discuss/interview-question/system-design/575186/Design-a-Parking-Spot-System>
<https://leetcode.com/discuss/interview-question/system-design/598634/Microsoft-or-Onsite-or-System-Design-or-SDE-2>
[https://leetcode.com/discuss/interview-question/system-design/850712/System-Design-Amazon-2020-\(SDE-2\)](https://leetcode.com/discuss/interview-question/system-design/850712/System-Design-Amazon-2020-(SDE-2))
<https://leetcode.com/discuss/interview-question/system-design/765686/System-Design-Interview-Question%3A-Parking-Lot-or-Low-Level-Design>
<https://leetcode.com/discuss/interview-question/system-design/125260/Parking-Lots-Design>
33. Recommendation Engine <https://leetcode.com/discuss/interview-question/system-design/124565/Design-Netflix-recommendation-engine>
34. Smart voice assistant like siri, alexa
<https://leetcode.com/discuss/interview-question/system-design/124566/Design-AlexaSiriGoogle-Home-Architecture>
<https://leetcode.com/discuss/interview-question/system-design/848252/Amazon-System-Design>
35. Nearest store location, another variation of topk
<https://leetcode.com/discuss/interview-question/system-design/124567/Nearest-Store-Locators>
<https://leetcode.com/discuss/interview-question/system-design/533061/How-to-implement-nearest-location-kind-of-functionality-in-a-google-map-type-application>
<https://leetcode.com/discuss/interview-question/system-design/154172/Design-google-map-database>
36. Job scheduling system <https://leetcode.com/discuss/interview-question/system-design/124697/Walmartlabs-onsite>
<https://leetcode.com/discuss/interview-question/system-design/124786/Google-Scheduling-Job-Involving-both-RAM-and-CPU>
<https://leetcode.com/discuss/interview-question/system-design/692996/Microsoft-System-Design-Please-help>
<https://leetcode.com/discuss/interview-question/system-design/553563/Google-or-Distributed-System-or-Performance>
<https://leetcode.com/discuss/interview-question/system-design/344524/Amazon-or-Design-a-Job-Task-Scheduler>
<https://leetcode.com/discuss/interview-question/system-design/124672/Implement-a-task-scheduler>
37. Elevator system
<https://leetcode.com/discuss/interview-question/system-design/149264/Design-an-Elevator-system>
38. Malware detection system <https://leetcode.com/discuss/interview-question/system-design/1019028/FB-or-System-Design-or-Multi-Engine-Malware-Analyzer>

<https://leetcode.com/discuss/interview-question/system-design/150610/Design-a-malware-detection-system>

39. Garbage collector

40. Google docs <https://leetcode.com/discuss/interview-question/system-design/148187/System-Design-or-Google-Docs>

<https://leetcode.com/discuss/interview-question/system-design/148187/System-Design-or-Google-Docs>

<https://leetcode.com/discuss/interview-question/system-design/208207/Design-a-Google-Sheet-System>

<https://leetcode.com/discuss/interview-question/system-design/349669/Google-SWE-L5-or-Onsite-or-Design-Google-Docs-Versioning-System>

<https://leetcode.com/discuss/interview-question/system-design/322448/Content-Management-System-Design>

<https://leetcode.com/discuss/interview-question/system-design/194402/Design-a-file-sharing-system>

41. Ecommerce Price checker system [https://leetcode.com/discuss/interview-question/system-design/140742/E-commerce-\(Amazon\)Website-looking-into-other-competitor-Website-products-prices-and-update](https://leetcode.com/discuss/interview-question/system-design/140742/E-commerce-(Amazon)Website-looking-into-other-competitor-Website-products-prices-and-update)

42. Notification system <https://leetcode.com/discuss/interview-question/system-design/138097/Design-Notification-Service-for-Amazon-Alexa>

43. Online ludo game

44. metric monitoring service

45. Ecommerce site ,Shopping cart, product catalog, payment gateway

[https://leetcode.com/discuss/interview-question/system-design/211415/Interview-Question-Ecommerce-System-design-\(-Eg-%3A-Amazon-\)%3A-Concurrency-issues-handling](https://leetcode.com/discuss/interview-question/system-design/211415/Interview-Question-Ecommerce-System-design-(-Eg-%3A-Amazon-)%3A-Concurrency-issues-handling)

<https://leetcode.com/discuss/interview-question/system-design/589546/Amazon-or-System-Design-or-Amazon-Order-System>

<https://leetcode.com/discuss/interview-question/system-design/675539/System-Design-question-asked-in-interview>

<https://leetcode.com/discuss/interview-question/system-design/666792/Microsoft-or-System-design-or-Please-help>

<https://leetcode.com/discuss/interview-question/system-design/1124722/System-Design-or-Shopping-Cart-or-Payment-Gateway-or-Product-Catalog>

<https://leetcode.com/discuss/interview-question/system-design/886390/Design-Recommendation-API-or-Akamai-Interview>

<https://leetcode.com/discuss/interview-question/system-design/776927/Design-an-accountpayment-system>

<https://leetcode.com/discuss/interview-question/system-design/706038/System-Design-Payment-System-Wallet-system-Payment-gateway>

46. Seller summary page <https://leetcode.com/discuss/interview-question/system-design/124612/Phone-Interview-Question%3A-Design-an-Seller-Summary-Page>

47. Customer who bought this also bought <https://leetcode.com/discuss/interview-question/system-design/124557/Amazon's-%22Customers-who-bought-this-item-also-bought%22-recommendation-system>

48. Distributed key value store, [https://leetcode.com/discuss/interview-question/system-design/1120468/Design-Assignment-or-Implement-a-distributed-Key-Value-\(KV\)-store-or-SE-Role-Avalara](https://leetcode.com/discuss/interview-question/system-design/1120468/Design-Assignment-or-Implement-a-distributed-Key-Value-(KV)-store-or-SE-Role-Avalara)

<https://leetcode.com/discuss/interview-question/system-design/747591/Amazon-or-Onsite-or-System-design-or-Please-help>

49. Facebook live commenting

<https://leetcode.com/discuss/interview-question/system-design/583184/FBInstagram-'Live-Comments'-System-design>

50. Facebook status search
51. Image editing (asked in fb 2021) <https://leetcode.com/discuss/interview-question/system-design/1077411/Facebook-or-Onsite-2021-or-System-Design-or-Design-image-editing>
52. File download application system <https://leetcode.com/discuss/interview-question/system-design/1071562/Design-a-File-Download-Application-System>
53. Proximity server <https://leetcode.com/discuss/interview-question/system-design/923677/Facebook-or-System-Design>
54. Top N songs, another top k problem
<https://leetcode.com/discuss/interview-question/system-design/124702/Design-a-service-to-calculate-the-top-k-listened-songs-in-past-24-hours>
<https://leetcode.com/discuss/interview-question/system-design/243604/Design-a-real-time-dashboard-showing-the-most-played-songs>
55. Privacy setting at facebook
56. Distributed configuration management system
57. Design gmail <https://leetcode.com/discuss/interview-question/system-design/1014986/Google-or-Onsite-or-System-Design%3A-Design-an-Email-system-like-GMAIL>
58. News reading feature in alexa <https://leetcode.com/discuss/interview-question/system-design/1014181/Amazon-or-System-Design-or-SDE2>
59. Ads click visualisation system <https://leetcode.com/discuss/interview-question/system-design/1002923/Facebook-or-Online-or-Real-time-data-visualization-for-ads-clicks>
60. IoT devices management system <https://leetcode.com/discuss/interview-question/system-design/974890/Design-a-system-for-management-of-IOT-devices>
61. Timer service <https://leetcode.com/discuss/interview-question/system-design/973207/System-Design-or-Timer-service>
62. Service monitoring and alerting system like pagerduty, azure monitor etc
<https://leetcode.com/discuss/interview-question/system-design/958919/System-Design-Interview-or-Service-Health-Monitoring-and-Alerting-Service>
<https://leetcode.com/discuss/interview-question/system-design/287678/Design-a-monitoring-or-analytics-service-like-Datadog-or-SignalFx>
63. Load balancer <https://leetcode.com/discuss/interview-question/system-design/943352/Facebook-or-E5-System-Design-Interview-Question-or-Menlo-Park>
64. Design underground system
65. Leader board table design <https://leetcode.com/discuss/interview-question/system-design/892083/Leaderboard-table-system-design-for-online-game>
66. Slot booking system for playarena etc <https://leetcode.com/discuss/interview-question/system-design/880581/Event-Booking-for-playarenas-Low-level-design>
<https://leetcode.com/discuss/interview-question/system-design/423613/Amazon-or-Phone-Screen-or-Design-Restaurant-Reservation-System>
67. Food delivery app <https://leetcode.com/discuss/interview-question/system-design/874074/Food-Delivery-App-or-Low-Level-Design-or-Interview-Question>
68. Online gaming lobby service <https://leetcode.com/discuss/interview-question/system-design/874074/Food-Delivery-App-or-Low-Level-Design-or-Interview-Question>
69. URL fishing varifier <https://leetcode.com/discuss/interview-question/system-design/896312/Google-system-design>
70. Design whatsapp/instagram story

<https://leetcode.com/discuss/interview-question/system-design/388222/Snapchat-or-System-Design-or-Instagram-Story-Feature>

71. File sharing with collaborative editing <https://leetcode.com/discuss/interview-question/system-design/838085/File-sharing-service-with-collaborative-editing-or-Amazon>

<https://leetcode.com/discuss/interview-question/system-design/824659/Intuit-or-Long-Poll-vs-Web-socket-vs-Server-send-Events>

72. Stack overflow tags <https://leetcode.com/discuss/interview-question/system-design/838025/Design-a-tagging-system-like-tags-used-in-stack-overflow>
<https://leetcode.com/discuss/interview-question/system-design/307558/Design-Stack-Overflow>

73. Tinyurls <https://leetcode.com/discuss/interview-question/system-design/838012/URL-Shortener-or-MD5-or-How-to-deal-with-collisions-or-FinTech-startup>

74. Github like cloud repo <https://leetcode.com/discuss/interview-question/system-design/837383/System-design-of-code-repository-like-github>

75. Job posting site <https://leetcode.com/discuss/interview-question/system-design/811840/Job-listing-storage-and-search>

76. S3/cloud object store <https://leetcode.com/discuss/interview-question/system-design/811503/System-design-Object-store-design-like-S3GCS>

77. Celebrity timeline generation <https://leetcode.com/discuss/interview-question/system-design/810561/Timeline-generation-for-celebrities-or-System-Design-or-Google>

78. Kindle service

79. Bidding system <https://leetcode.com/discuss/interview-question/system-design/792060/Bidding-System%3A-System-Design-Interview>

80. Billing system - asked in fb interview

81. RPC system for client server comm <https://leetcode.com/discuss/interview-question/system-design/790034/Client-Server-Communication%3A-System-Design-Interview>

82. Autonomous driving system <https://leetcode.com/discuss/interview-question/system-design/789961/Design-a-cloud-based-simulationvisualization-platform-for-a-self-driving-cars-company>

83. Github code search <https://leetcode.com/discuss/interview-question/system-design/789015/Github-%3A-Design-search-feature-in-Github-scale-code-repository>

84. Car showroom <https://leetcode.com/discuss/interview-question/system-design/785960/Amazon-System-Design-Question>

85. Airport boarding gate security <https://leetcode.com/discuss/interview-question/system-design/785960/Amazon-System-Design-Question>

86. Social graph <https://leetcode.com/discuss/interview-question/system-design/782906/Design-a-social-graph>

87. Place of interest <https://leetcode.com/discuss/interview-question/system-design/777945/Design-a-system-to-source-store-and-display-places-of-interest>

88. Tinder <https://leetcode.com/discuss/interview-question/system-design/774870/Tinder-System-Design-or-Online-Dating-App-System-Design>

89. Grocery store <https://leetcode.com/discuss/interview-question/system-design/769578/Amazon-orSystem-Design-or-Amazon-Go-or-suggestion-on-solution-welcome>

<https://leetcode.com/discuss/interview-question/system-design/467655/Amazon-Onsite-or-System-Design-Pickup-Delivery-System-For-Groceries>

90. Display ads <https://leetcode.com/discuss/interview-question/system-design/761814/Design-number-of-ads-to-show-to-users-on-a-google-search>
91. Add badge to peoples spotify account <https://leetcode.com/discuss/interview-question/system-design/748408/How-to-design-a-system-to-add-badges-to-people's-Spotify-account>
92. Xml to json conversation <https://leetcode.com/discuss/interview-question/system-design/743624/System-design-question%3A-Amaozon-SDE2%3A-Large-xml-files-to-json-conversion>
93. Ocr web app <https://leetcode.com/discuss/interview-question/system-design/741676/System-Design%3A-OCR-web-app>
94. High scale otp generation system <https://leetcode.com/discuss/interview-question/system-design/728464/Microsoft-or-Onsite-or-Modify-an-OTP-generation-system-to-handle-more-requests>
95. Distributed counter <https://leetcode.com/discuss/interview-question/system-design/685310/Microsoft-virtual-or-Design-distributed-counter>
<https://leetcode.com/discuss/interview-question/system-design/277606/Design-a-performance-counter>
96. Scan for viruses in uploaded file <https://leetcode.com/discuss/interview-question/system-design/659875/Design-a-system-where-client-can-upload-a-file-and-viruses-need-to-be-scanned>
97. Log processing at scale <https://leetcode.com/discuss/interview-question/system-design/622704/Design-a-system-to-store-and-retrieve-logs-for-all-of-eBay>
98. London travel card system [https://leetcode.com/discuss/interview-question/system-design/617408/Marshall-Wace-or-Onsite-or-How-would-you-design-Oyster-\(London-Travel-Card-system\)tion](https://leetcode.com/discuss/interview-question/system-design/617408/Marshall-Wace-or-Onsite-or-How-would-you-design-Oyster-(London-Travel-Card-system)tion)
99. Payment system for newyork MTA <https://leetcode.com/discuss/interview-question/system-design/305388/Design-a-transportation-payment-System>
100. Ad click counter <https://leetcode.com/discuss/interview-question/system-design/584458/Facebook-or-System-Design-or-Ad-Click-Counter>
101. Design slack <https://leetcode.com/discuss/interview-question/system-design/582975/Design-Slack>
<https://leetcode.com/discuss/interview-question/system-design/339849/System-Design-or-Slack>
102. Wikipedia <https://leetcode.com/discuss/interview-question/system-design/574872/Wikipedia-or-DBsystem-design-thoughts>
<https://leetcode.com/discuss/interview-question/system-design/174380/Uber-design-question-Design-Wikepeida>
103. ML related system design <https://leetcode.com/discuss/interview-question/system-design/566057/Machine-Learning-System-Design-%3A-A-framework-for-the-interview-day>
104. Windows update <https://leetcode.com/discuss/interview-question/system-design/560512/System-Design-Question>
105. People also searched for <https://leetcode.com/discuss/interview-question/system-design/559481/Amazon-or-System-design-or-SDE-2-India> smiliar to linkedin's you may also know
106. Cashback processing system <https://leetcode.com/discuss/interview-question/system-design/543041/Design-cashback-processing-system>
107. Pub sub arch
108. Google or amazob book preview <https://leetcode.com/discuss/interview-question/system-design/538295/Design-Google-Books-preview-Amazon-Books-look-inside>

109. Design copy right detection <https://leetcode.com/discuss/interview-question/system-design/530031/FAANG-Interview-Question-Design-a-copyright-detection-system>
110. Reddit <https://leetcode.com/discuss/interview-question/system-design/469900/Netflix-or-System-Design-Web-App-Like-Reddit>
111. Facebook nearby friends <https://leetcode.com/discuss/interview-question/system-design/430926/Design-Nearby-Friends>
112. Google photos home page <https://leetcode.com/discuss/interview-question/system-design/398523/System-Design-Google-photos-homepage>
<https://leetcode.com/discuss/interview-question/system-design/396949/System-Design-Google-Photos>
113. Image upload system <https://leetcode.com/discuss/interview-question/system-design/391183/Ebay-System-Design-Question>
<https://leetcode.com/discuss/interview-question/system-design/390503/Google-or-System-Design>
114. Facebook translator service <https://leetcode.com/discuss/interview-question/system-design/386322/Design-a-translator-service-for-facebook>
<https://leetcode.com/discuss/interview-question/system-design/318811/Google-or-System-design-or-Design-a-translation-service-like-Google-Translate>
115. System for health care data <https://leetcode.com/discuss/interview-question/system-design/368245/Design-Service-to-Interface-with-Healthcare-Data>
116. Health score app <https://leetcode.com/discuss/interview-question/system-design/366754/Amazon-or-System-Design-for-health-score-app>
117. Railway reservation system <https://leetcode.com/discuss/interview-question/system-design/364965/Railway-Reservation-System>
118. Treadmill system <https://leetcode.com/discuss/interview-question/system-design/362168/Google-or-Onsite-or-Tread-Mill-System-Design>
119. Addressed of entire planet <https://leetcode.com/discuss/interview-question/system-design/341980/Amazon-or-System-Design-or-System-to-capture-unique-addresses-in-the-entire-world>
120. Gofundme <https://leetcode.com/discuss/interview-question/system-design/336089/System-Design-or-GoFundMe>
121. Shipping fulfillment <https://leetcode.com/discuss/interview-question/system-design/320719/Design%3A-Scalable-Shipping-Fulfillment-Center>
122. Flight search API <https://leetcode.com/discuss/interview-question/system-design/309853/JSON-structure-for-Flight-search-API>
123. Splitwise <https://leetcode.com/discuss/interview-question/system-design/306519/System-Design-or-Splitwise>
124. Google calendar <https://leetcode.com/discuss/interview-question/system-design/305654/System-Design-or-Google-Calendar>
125. Fb popular/trending pages <https://leetcode.com/discuss/interview-question/system-design/305505/Design-a-most-populartrending-profiles-page>
126. Courier service <https://leetcode.com/discuss/interview-question/system-design/301423/Design-a-UPS-style-mail-delivery-system>
127. Hr portal <https://leetcode.com/discuss/interview-question/system-design/289092/Design-an-HR-web-portal-for-Amazon's-recruiting-team>
128. Outlook recurring meeting <https://leetcode.com/discuss/interview-question/system-design/286891/Design-Outlook-recurring-meeting-system-with-variable-input>
129. Communication system for ecom sites <https://leetcode.com/discuss/interview-question/system-design/286457/Design-a-communication-platform>.

130. Imdb <https://leetcode.com/discuss/interview-question/system-design/270416/Design-a-movies-reviews-aggregator-system>
131. Realtime event aggregator <https://leetcode.com/discuss/interview-question/system-design/270412/Design-a-Real-Time-Event-Aggregation-System>
132. Gpay <https://leetcode.com/discuss/interview-question/system-design/270406/Design-a-Payment-System-like-Google-Pay>
133. Top shared post <https://leetcode.com/discuss/interview-question/system-design/258398/Design-top-shared-post-system-in-5mins1-hour1-day1-week>
134. Market place analytics <https://leetcode.com/discuss/interview-question/system-design/227797/System-Design-E-Commerce-Marketplace-analytics>
135. Top 10 most liked articles <https://leetcode.com/discuss/interview-question/system-design/225609/Design-system-which-will-show-top-10-most-liked-articles-within-1524-hours>.
136. Auth for multi tenant <https://leetcode.com/discuss/interview-question/system-design/225331/Design-authentication-system-to-multi-tenant-environment>
<https://leetcode.com/discuss/interview-question/system-design/202958/Multi-Tenant-SaaS-Architecture>
137. Design Changefeed <https://leetcode.com/discuss/interview-question/system-design/208888/Design-a-system-to-keep-track-of-changes-in-an-SQL-database>
138. Track runners in marathon <https://leetcode.com/discuss/interview-question/system-design/200342/Bloomberg%3A-Implement-a-system-to-track-runners-in-a-marathon>
139. Hotel booking page this many people visiting
<https://leetcode.com/discuss/interview-question/system-design/163204/Design-%22How-Many-people-currently-viewing-the-property%22-for-a-E-Commerce-Hotel-Booking-Site>
140. Text line editor <https://leetcode.com/discuss/interview-question/system-design/124679/Implement-a-Text-Line-Editor>
141. Location sharing service <https://leetcode.com/discuss/interview-question/system-design/124673/Design-a-Location-Sharing-Android-Application>
142. Build system <https://leetcode.com/discuss/interview-question/system-design/124807/Design-a-build-system>
143. Hourly backup from mobile phone <https://leetcode.com/discuss/interview-question/system-design/124792/Design-a-system-that-can-handle-hourly-backups-for-mobile-phones>
144. Google help system <https://leetcode.com/discuss/interview-question/system-design/125191/Design-the-Google-help-system>

Other interesting post about specific component design

1. <https://leetcode.com/discuss/interview-question/system-design/136140/Write-a-class-which-is-hard-to-test>
2. Interesting variation of ecart problem where first one who click buys will buy
<https://leetcode.com/discuss/interview-question/system-design/498895/Startup-interview-or-Designing-tricky-e-shop>

3. Back of envelope calculation <https://leetcode.com/discuss/interview-question/system-design/357656/Experience-with-back-of-the-envelope-calculations>
4. Drawing tool <https://leetcode.com/discuss/interview-question/system-design/1148896/System-Design-Drawing-Tool-Recos>
5. Very good list <https://leetcode.com/discuss/general-discussion/670355/Experienced-Interview-Preparation-Guide-All-Resources>
6. [https://leetcode.com/discuss/interview-question/1002218/Facebook-or-Google-or-Top-System-Design-Interview-Questions-\(Part-1\)](https://leetcode.com/discuss/interview-question/1002218/Facebook-or-Google-or-Top-System-Design-Interview-Questions-(Part-1))
7. <https://github.com/donnemartin/system-design-primer>
8. <https://github.com/binhnguyennus/awesome-scalability>
9. Json parser <https://leetcode.com/discuss/interview-question/system-design/1052608/Design-a-JSON-Parser-or-SDE2>
10. <https://leetcode.com/discuss/interview-question/system-design/1043657/MakeMyTrip-Backend-Developer-or-System-DesignMultithreading>
11. [https://leetcode.com/discuss/interview-question/system-design/1042229/Facebook-or-Google-or-Top-System-Design-Interview-Questions-\(Part-2\)](https://leetcode.com/discuss/interview-question/system-design/1042229/Facebook-or-Google-or-Top-System-Design-Interview-Questions-(Part-2))
12. <https://leetcode.com/discuss/interview-question/system-design/1038585/How-do-you-scale-up-an-Application-to-serve-thousands-of-request-per-second>
13. User data access policy design <https://leetcode.com/discuss/interview-question/system-design/895268/Google-or-System-Design>
14. <https://leetcode.com/discuss/interview-question/system-design/829466/Amazon-or-Phone-or-Seattle-or-Column-Store-vs-Row-Store>
15. <https://leetcode.com/discuss/interview-question/system-design/808216/Phone-book-and-search> - suffix tree
16. <https://hackernoon.com/scaling-websockets-9a31497af051>
17. Online whiteboard drawing like draw.io
18. <https://leetcode.com/discuss/interview-question/system-design/799474/Virtual-onsite-at-DocuSign>
19. <https://leetcode.com/discuss/interview-question/system-design/795890/How-many-transactions-does-Oracle-DB-handle-Read-and-write>
20. How to store Recently viewed item <https://leetcode.com/discuss/interview-question/system-design/775139/Amazon-System-Design-customer's-recently-viewed-items>
21. Depth an interviewer can go <https://leetcode.com/discuss/interview-question/system-design/773980/Watch-this-before-System-Design-Interview-the-details-an-interviewer-can-go-to-evaluate-candidate>
22. Privacy api <https://leetcode.com/discuss/interview-question/system-design/727474/System-design-critique-request-for-below-question>
23. LLD <https://leetcode.com/discuss/interview-question/system-design/692383/Google-or-Onsite-or-Design-a-organization-pharmacy-shop-with-managers>
24. T9 predicitive system <https://leetcode.com/discuss/interview-question/system-design/685338/Microsoft-or-Onsite-or-Design-the-T9-predictive-text-algorithm-and-system>
25. Storsge of millions of subscriber [https://leetcode.com/discuss/interview-question/system-design/680047/How-will-you-store-millions-of-subscribers-list-\(assume-it-as-email-id\)](https://leetcode.com/discuss/interview-question/system-design/680047/How-will-you-store-millions-of-subscribers-list-(assume-it-as-email-id))
26. Flipkart warehouse portal <https://leetcode.com/discuss/interview-question/system-design/663037/Amazon-SystemDesign-Flipkart-Suggestions-Design-Warehouse-Portal>

27. <https://leetcode.com/discuss/interview-question/system-design/632537/Design-an-algorithm-to-efficiently-transfer-required-bytes-of-data-to-a-single-PC-on-the-network>.
28. <https://leetcode.com/discuss/interview-question/system-design/581804/Facebook-System-Design-Preparation> - different type of design interviews at facebook
29. Count current active user on the page, <https://leetcode.com/discuss/interview-question/system-design/557603/Postman-or-OA-or-System-Design>
<https://leetcode.com/discuss/interview-question/system-design/532889/Design-a-counter-for-a-website-which-tells-how-many-visits-happened-to-that-website>
30. Csv parsing at scale <https://leetcode.com/discuss/interview-question/system-design/545664/CSV-parsing-at-scale>
31. API to get best selling book <https://leetcode.com/discuss/interview-question/system-design/535162/Design-read-api-to-get-best-selling-books>
32. Fb mutual friend <https://leetcode.com/discuss/interview-question/system-design/533810/FB-API-System-design>
33. <https://leetcode.com/discuss/interview-question/system-design/532089/Update-System-design-of-Amazon-to-handle-10x-times-more-traffic-than-what-it-currently-receives>
34. <https://leetcode.com/discuss/interview-question/system-design/513374/FAANG-system-design-interview-question>
35. Sql scalability <https://leetcode.com/discuss/interview-question/system-design/507833/SQL-Scalability>
36. Backup from dc1 to dc2 <https://leetcode.com/discuss/interview-question/system-design/502522/System-design-to-backup-datacenter1-to-datacenter-2>
37. Order within next 1 hour to get early delivery feature
<https://leetcode.com/discuss/interview-question/system-design/499558/Amazon-System-Design-Question>
38. <https://leetcode.com/discuss/interview-question/484956/design-airport-luggage-handling-system>
39. [https://leetcode.com/discuss/interview-question/system-design/483959/Google-onsite-\(Theoretical-System-Design\)](https://leetcode.com/discuss/interview-question/system-design/483959/Google-onsite-(Theoretical-System-Design))
40. <https://leetcode.com/discuss/interview-question/system-design/421969/Twitch-or-System-Design-Onsite-or-Design-Twitch-Analytics-Use-Case>
41. Good discussion on designing SAAS <https://leetcode.com/discuss/interview-question/system-design/385363/Design-Task-Executor-as-Saas>
42. Track down bad request <https://leetcode.com/discuss/interview-question/system-design/313117/Design-a-system-capable-of-tracking-bad-request-down-quickly>
43. Facebook <https://leetcode.com/discuss/interview-question/system-design/311825/Facebook-System-design>
44. Flight landing with onerunway <https://leetcode.com/discuss/interview-question/system-design/303745/Handling-flight-landing-requests-with-one-runway>
45. Good read on data modelling <https://leetcode.com/discuss/interview-question/system-design/295671/Amazon-or-Data-Engineer-Role-or-Database-Design-Question>
46. Traffic light controller <https://leetcode.com/discuss/interview-question/system-design/291233/Design-a-traffic-light-controller>
47. Live streaming view count <https://leetcode.com/discuss/interview-question/system-design/284232/Livestreaming-view-count>.
48. Random number generator for a slot mschine <https://leetcode.com/discuss/interview-question/system-design/281762/Microsoft-or-Design-a-random-number-generator-for-a-slot-machine>

49. Company badge system <https://leetcode.com/discuss/interview-question/system-design/279371/Design-a-badge-system>
50. API gateway with graphql <https://leetcode.com/discuss/interview-question/system-design/255282/Design-API-gateway>
51. Good read on scaling <https://leetcode.com/discuss/interview-question/system-design/250803/Large-amount-of-query-request-in-seconds>.
52. <https://leetcode.com/discuss/interview-question/system-design/249988/How-to-Transfer-1-GB-of-data-from-one-microservice-to-another>
53. <https://leetcode.com/discuss/interview-question/system-design/234311/Celebrity-Twitter-hack>
54. Amazon locker <https://leetcode.com/discuss/interview-question/system-design/233869/Design-Amazon-Locker-system>
55. Upload large file <https://leetcode.com/discuss/interview-question/system-design/224398/Algorithm-for-upload-large-file>
56. Currency civersation <https://leetcode.com/discuss/interview-question/system-design/216912/Design-an-app-that-converts-USD-to-another-country-currency>.
57. Geohash <https://leetcode.com/discuss/interview-question/system-design/203364/How-to-retrieve-addresses-based-on-Latitude-and-Longitude>
<https://leetcode.com/discuss/interview-question/system-design/124568/How-to-design-a-system-to-retrieve-address-information-longitude-and-latitude-information>.
58. Strategy to update sattelite firmware <https://leetcode.com/discuss/interview-question/system-design/198761/Updating-Satellite-Frimware>
59. Design card deck <https://leetcode.com/discuss/interview-question/system-design/194663/Design-a-class-that-represents-a-deck-of-cards>
60. Send large files [https://leetcode.com/discuss/interview-question/system-design/193953/Distribute-binary-file-\(daily\)-for-thousands-of-servers](https://leetcode.com/discuss/interview-question/system-design/193953/Distribute-binary-file-(daily)-for-thousands-of-servers)
61. Extract json from a text file <https://leetcode.com/discuss/interview-question/system-design/191944/How-to-extract-JSON-object-from-a-text-file-of-size-100-GB>
62. <https://leetcode.com/discuss/interview-question/system-design/177823/How-to-track-overall-activity-time-of-every-user-efficiently>
63. Design video pause functionality <https://leetcode.com/discuss/interview-question/system-design/177327/Video-pause-functionality-for-multiple-devices>
64. <https://leetcode.com/discuss/interview-question/system-design/172809/Sync-front-end-with-back-end-in-realtime>
65. Push/pull API for producer consumer <https://leetcode.com/discuss/interview-question/system-design/158701/Design-push-and-pull-APIs-for-a-producer-consumer-system>
66. HA for webproxy <https://leetcode.com/discuss/interview-question/system-design/134797/HA-for-HTTP-proxy>
67. Alarm app <https://leetcode.com/discuss/interview-question/system-design/133426/Microsoft-Mobile-app-for-alarm>
68. Snake and ladder <https://leetcode.com/discuss/interview-question/system-design/132140/Design-online-multiplayer-snakeandladder-game>
69. [https://leetcode.com/discuss/interview-question/system-design/124858/First-non-repeating-word-in-a-file-File size-can-be-100GB](https://leetcode.com/discuss/interview-question/system-design/124858/First-non-repeating-word-in-a-file-File-size-can-be-100GB).
70. Store extremely large parse matrix <https://leetcode.com/discuss/interview-question/system-design/125306/How-to-store-extremely-large-sparse-matrices>
71. Implement mine swipper

OOP design <https://github.com/tssovi/grokking-the-object-oriented-design-interview>
<https://github.com/savitansh/SystemDesignInterview>

<https://leetcode.com/discuss/interview-question/system-design/943886/Facebook-Product-Design-Questions>

<https://github.com/checkcheckzz/system-design-interview>
GitHub - donnemartin/system-design-primer: Learn how to design large-scale systems. Prep for the system design interview. Includes Anki flashcards.

Drawing tool

<https://leetcode.com/discuss/interview-question/system-design/758105/Remote-system-design-diagram-drawing-tool>

Other informative posts

I hope it will be helpful if you have mentioned your overall experience.
If you are looking for interview perspective, You need to gain knowledge about distributed systems.
To start with, to simplify it,
For a Simple Application
Input --> Processing --> DataStore --> Processing/Output
For Distributed Systems [I wrote considering for System Design interview]
Following contains quick overlook about the list that need to have for basic understanding.

1. Input/Output
Communication[Rest, grpc, WebSocket, tcp/udp, webRTC], API Gateway, Proxy Server,
Load Balancing -- [Reverse Proxy]
Message Queue -- [Queue for sending/receiving information] Kafka/RabbitMQ
2. Processing
Computation -- Micro-service [Python, Springboot/Java, Go...], MapReduce
Service Discovery/Registry -- [Finding and Redirecting the load, Saga Pattern] -- Eg: Eureka
3. Datastore/Distributed Consensus
Cache -- [Simple Cache to reduce database hits] -- Eg: Redis
SQL -- [SQL database for Relational Data -- Transactions based like payments, Horizontal Partition/Vertical Partition, Sharding] Eg: MySQL
NOSQL -- [NOSQL for querying documents like product, product details, CAP] Eg: MongoDB/Apache Cassandra
Concurrency -- 2 Phase Commit, 3 Phase Commit, Saga Pattern[Choreography saga, Orchestrator Saga], Split Brain problem.
FileSystem -- Hadoop File System
4. Security
AAA -- Authentication, Authorisation, Auditing
Misc -- Logging/Notification

Once you have overall picture in your mind, you can start with YouTube videos like Gaurav Sen or Tech Dummies.

Product Design

The product design interview at Facebook will involve designing a product or API to support an end-user experience. Here's a list of concepts that Facebook recommends you review before your interview:

- Scalability
- Design patterns
- Data ownership
- Protocols
- Data formats
- Client-server design
- Designing for long term vs. complexity
- Accommodating possible product changes

Some example questions involve designing a product API or an email server.

System Design

The system design interview at Facebook will ask you to weigh design considerations for complex problems. Here's a list of concepts that Facebook recommends you review before your interview:

- Concurrency (threads, deadlock, starvation, consistency, coherence)
- Caching
- Database partitioning, replication, sharding, CAP Theorem
- Networking (IPC, TCP/IP)

Real-world performance (relative performance RAM, disk, your network, SSD)

- Availability and reliability (types of failures, failure units, how failures may manifest, mitigations, etc.)
- Data storage and data aggregation
- QPS capacity/machine estimation (back of the envelope estimates), byte size estimation

Some example questions involve architecting a video distribution system or designing a mobile image search client.