10 System Design Topics You Must Prepare for Tech Interviews

These are the 10 topics every developer must prepare for tech interviews to get ahead of 99% of candidates.

JAVINPAUL

DEC 21, 2023 • PAID

Hell guys, if you have attended technical interviews then you may know that the System Design part is one of the toughest to crack. I think why so? and the answer I found was that most of the developers are not familiar with essential System design topics or concepts and that's what I am going to share in this article.

As I have said before, System design interviews are a crucial part of the hiring process for software engineers and developers and you must prepare for it, leaving it to chance is not a good idea.

These interviews assess your ability to design scalable and efficient systems to solve real-world problems. To excel in system design interviews, it's essential to have a strong grasp of the fundamental concepts and principles.

In the past, I have shared several system design interview articles like API Gateway vs load balancer, Forward Proxy vs Reverse Proxy as well as common System Design problems In this article, we will explore ten essential system design topics or concepts that will help you prepare for your next interview and impress your potential employers.

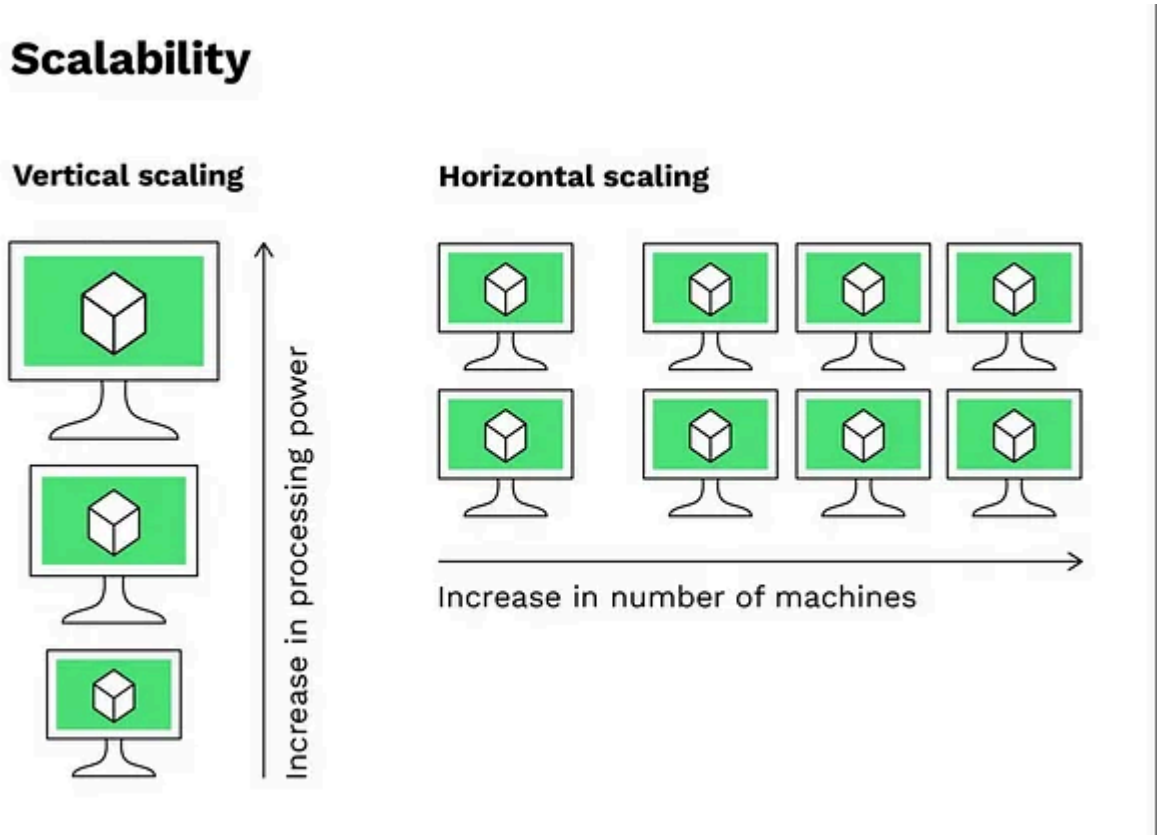
By the way, if you are *preparing for System design interviews* and want to learn System Design in-depth then you can also check out sites like **ByteByteGo**, **DesignGuru**, **Exponent**, **Educative,** and **Udemy** which have many great System design courses.

competition in system design interviews, it's crucial to have a solid understanding of key concepts that set you apart from other candidates.

By mastering these ten system design concepts, you can position yourself ahead of 99% of candidates and impress interviewers with your expertise.

1. Scalability

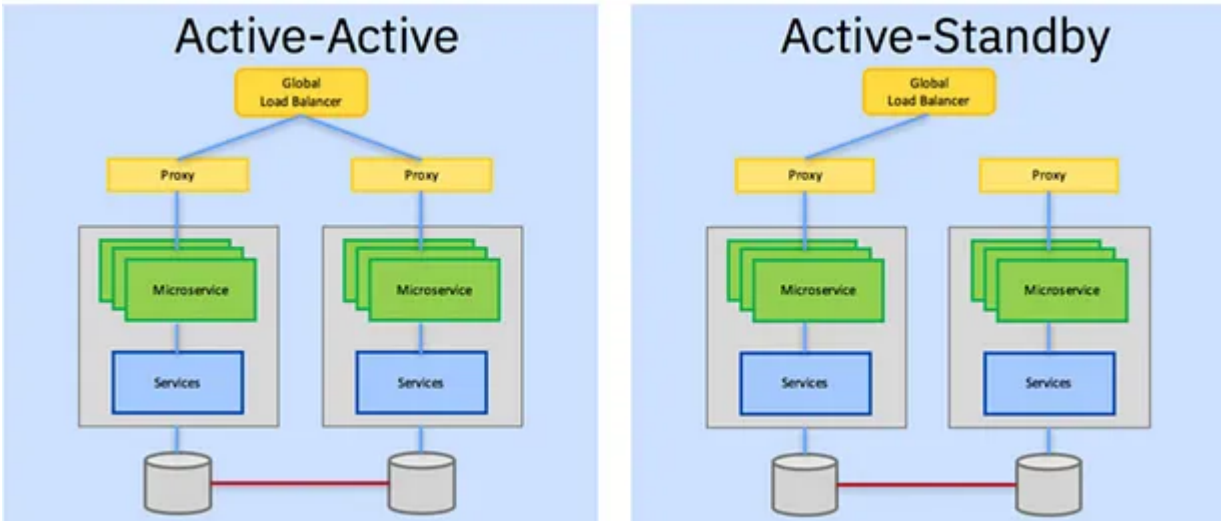
Scalability is crucial in system design as it ensures that a system can handle increasing loads and maintain performance. You should understand concepts like horizontal and vertical scaling, load balancing, and distributed systems to design scalable architectures. Those will help you with tech interviews.



2. Availability and Fault Tolerance

Designing fault-tolerant systems is essential to maintain availability even in the face of failures. As a candidate, you should learn about replication, redundancy, failover mechanisms, and fault tolerance techniques like backups, checkpoints, and error handling.

For high availability many companies create active-active or active-passive architecture as shown below, The best example is cloud computing companies like AWS and Azure who has data centers in different parts of the world, and if one goes down then server clients from others.

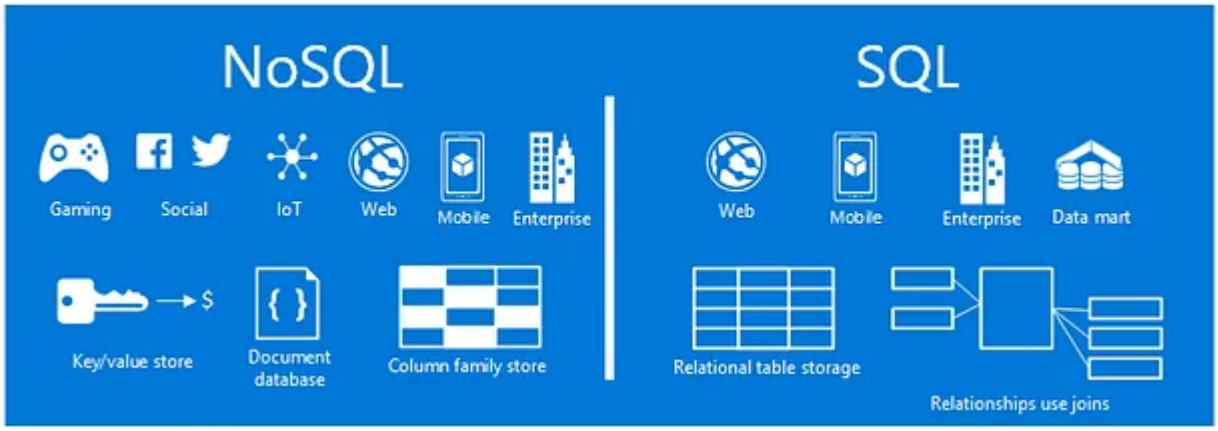


3. Data Storage and Databases

Different applications require different types of data storage. You should familiarize yourself with different types of databases like relational databases, NoSQL databases, key-value stores, and columnar databases. Understand their strengths, weaknesses, and use cases.

One of the common questions and concepts

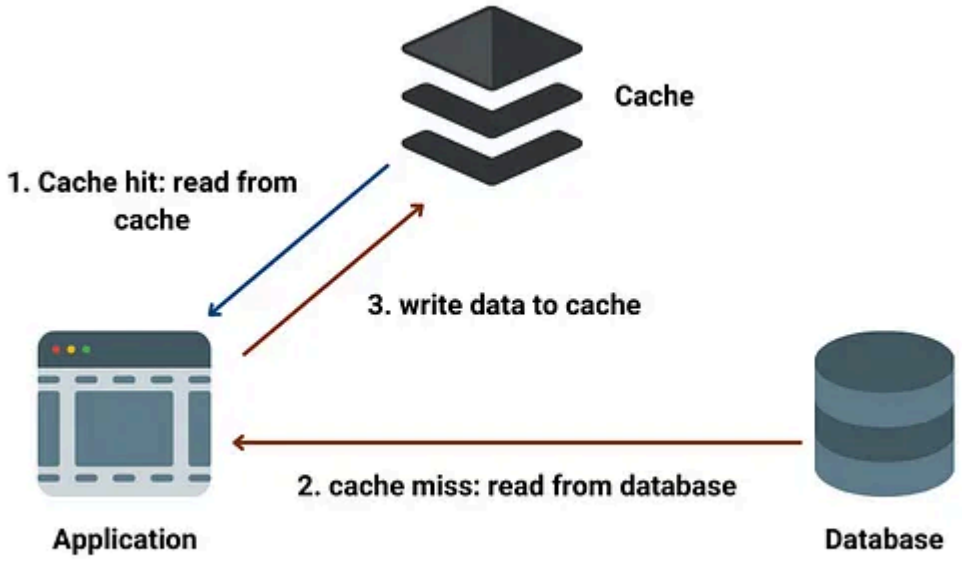
to understand SQL vs NoSQL as they often come during different questions, here is a nice diagram that highlights the difference between SQL and NoSQL



4. Caching

Caching helps improve performance by storing frequently accessed data closer to the users. In this topic, you should learn about caching techniques, caching strategies, cache eviction policies, and cache coherence to design efficient caching systems.

You should also get yourself familiar with things like cache hit and cache miss and here is a nice diagram for quick reference:



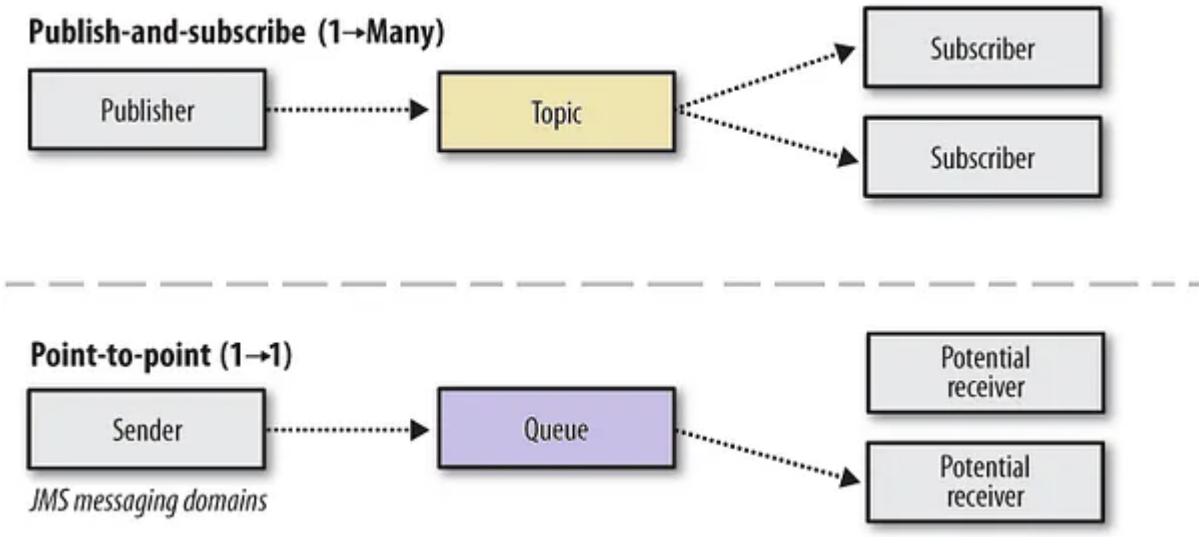
Caching

5. Message Queues and Event-driven Architecture

Message queues enable asynchronous communication between different components of a system. While preparing for this topic you should understand concepts like pub-sub (publish-subscribe) patterns, message brokers, event-driven architecture, and their applications in building scalable and loosely coupled systems.

In the past, I wrote about Apache Kafka vs ActiveMQ vs RabbitMQ which is a good starting point for learning about message brokers and Queue.

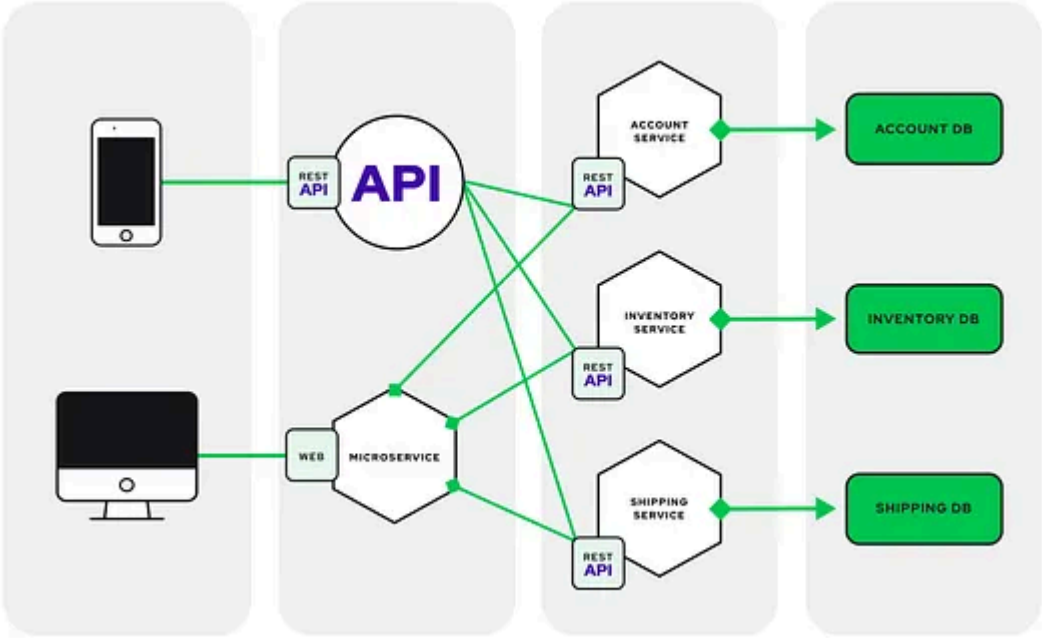
Here is a nice diagram showing Event Driven Architecture using the pub-sub model and point-to-point model :



6. System APIs and Microservices

This is another popular topic in System design interviews. Microservices architecture allows breaking down complex systems into smaller, independent services.

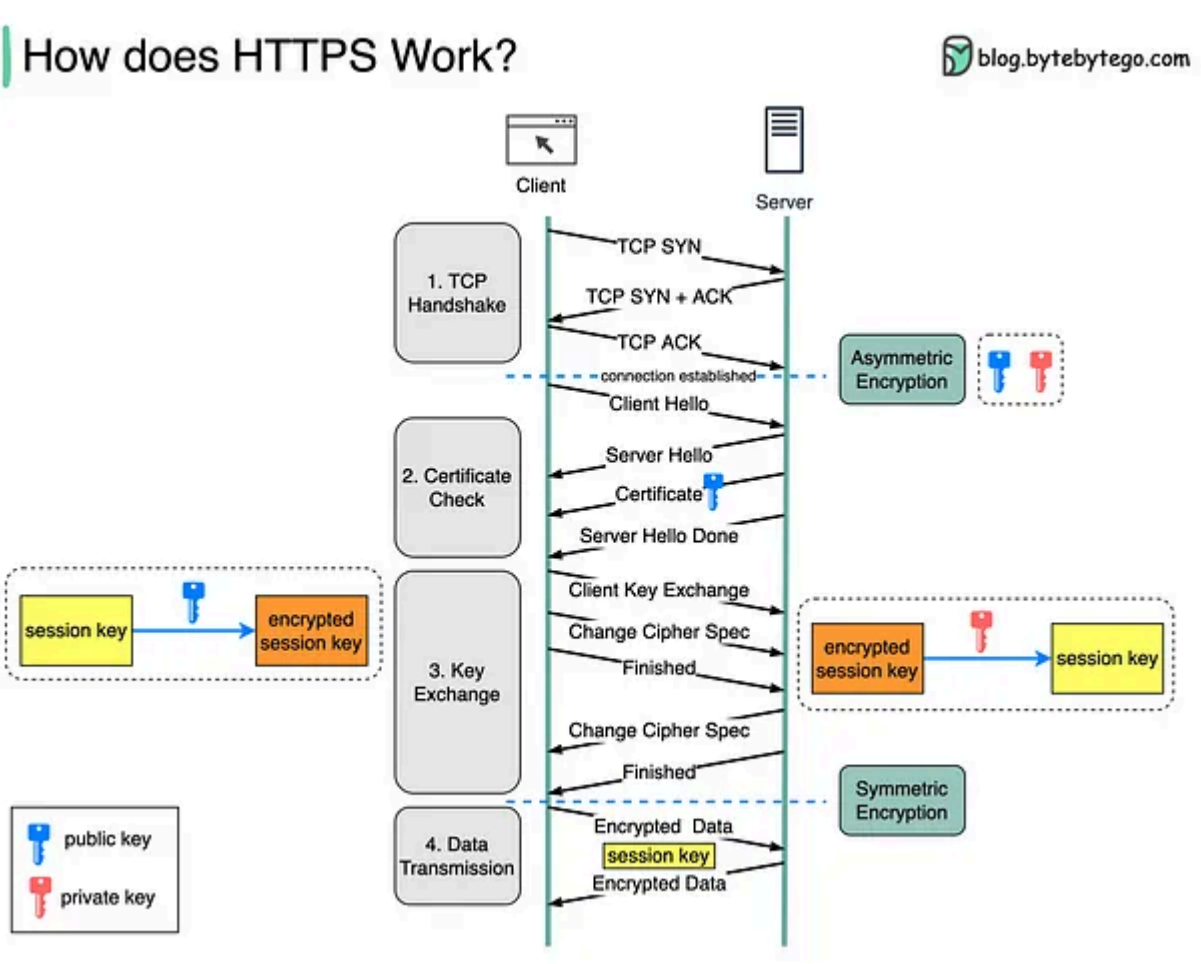
In this topic, you should learn about designing APIs, service discovery, inter-service communication, and managing dependencies to create robust and scalable microservices-based architectures.



7. Security

Security is a critical aspect of system design. While preparing for this topic you should spend time understanding common security threats, authentication and authorization mechanisms, encryption, and secure communication protocols like SSL/TLS. Be aware of best practices for securing data at rest and in transit.

Learning things like How HTTPS works is also a great question to start with. Here is a nice diagram from ByteByteGo to understand that:



image_credit — bytebytego

8. System Performance Optimization

Optimizing system performance is vital for providing a seamless user experience and this is also one thing that interviewers expect from you, especially from senior developers.

As part of this topic, you should learn about profiling, load testing, latency reduction techniques, and performance optimization strategies to identify and resolve bottlenecks in your system.

Learning about Database optimization using normalization and index as well as understanding how SQL queries work to optimize it is also a great topic to start with

9. Design Patterns

Design patterns offer proven solutions to recurring design problems. As a software engineer, you should familiarize yourself with design patterns like the Singleton, Observer, Builder, and Factory patterns.

You should also understand when and how to apply them to create scalable and maintainable systems. You should also learn about Microservices design patterns like SAGA, CQRS, Load Balancer, Circuit Breaker, DB Per Microservices, and API Gateway patterns.

10. Trade-offs and System Constraints

Nothing is perfect and compromise and trade-off is what you need in the real world to work. System design often involves making trade-offs based on specific constraints.

You should always consider factors like cost, time, available resources, and technology limitations. You should also understand how to make informed decisions based on these constraints without compromising the overall system design.

Knowing things like Big-O notation also helps when you talk about system design algorithms:

That's all about **10 System design topics you must prepare for tech interviews.** Mastering these ten essential system design concepts will significantly enhance your performance in system design interviews.

By understanding scalability, fault tolerance, data storage, caching, message queues, microservices, security, performance optimization, design patterns, and trade-offs, you'll be well-prepared to tackle complex design problems and impress interviewers with your comprehensive knowledge.

By the way, if you are *preparing for System design interviews* and want to learn System Design in-depth then you can also check out sites like **ByteByteGo**, **DesignGuru**, **Exponent**, **Educative** and **Udemy** which have many great System design courses and if you need free system design courses you can also see the below article.

2 Likes · 1 Restack

← Previous

Next →

1 Comment

Write a comment...

Soma Soma's Substack Dec 24, 2023

Thanks

LIKE REPLY SHARE