10 Must Know Distributed System Patterns

Mahesh Saini · Follow
4 min read · Jun 14, 2023

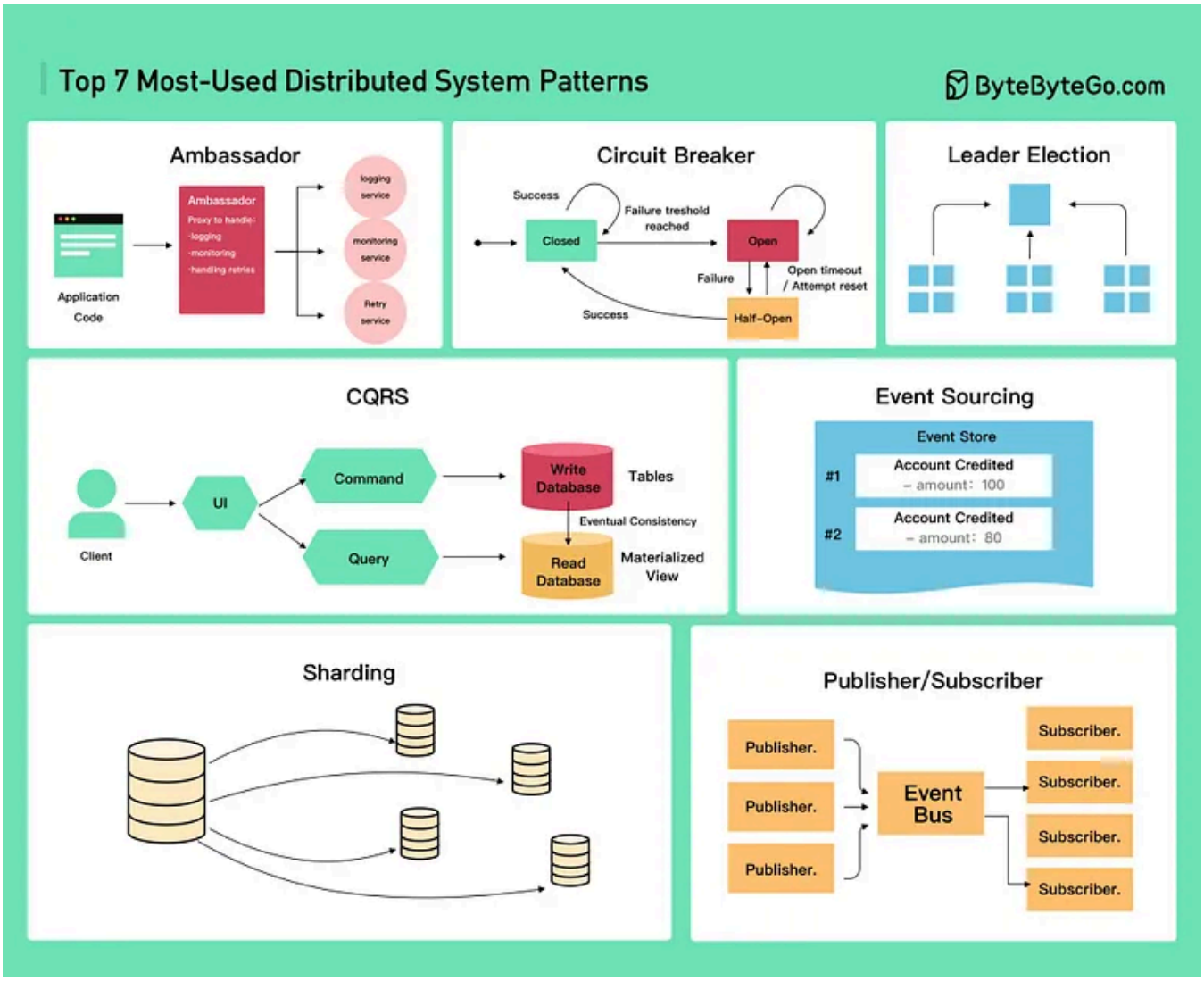


Image from — bytebytego.com

Distributed patterns can help us design more efficient and scalable systems, so let's dive right in.

1. Ambassador

- Picture yourself as a busy CEO with a personal assistant who handles all your appointments and communication.
- That's precisely what the Ambassador pattern does for our application. It acts as a go between for our app and the services it communicates with, offloading tasks like logging, monitoring, or handling retries.
- For instance, Kubernetes uses Envoy as an ambassador to simplify communication between services.
- The Ambassador pattern can help reduce latency, enhance security, and improve the overall architecture of your distributed systems.

2. Circuit Breaker

- Imagine a water bite bursting in your house. The first thing you would do is shut off the main valve to prevent further damage.
- The circuit breaker pattern works similarly, preventing cascading failures in distributed systems. When a service becomes unavailable, the circuit breaker stops requests allowing it to recover.
- Netflix Hystrix library uses this pattern. It ensures a more resilient system.
- Now this pattern can be particularly useful when dealing with microservices or cloud based applications where failures are more likely to occur.

3. Bulk Head

- In software architecture, the Bulkhead pattern involves dividing the system into separate compartments, or "bulkheads," where each compartment contains a set of resources or services. By isolating these compartments, failures or overloads in one compartment are contained within that compartment and do not propagate to other parts of the system.
- It is especially useful in distributed systems where failures or performance issues in one component can potentially affect other components.

4. CQRS or Command Query Responsibility Segregation

- CQRS is having a restaurant with separate lines for ordering food and picking up orders by separating the command or write operations from the query or read operations.

- We can scale and optimize each independently. An e-commerce platform might have high read requests for product listings, but fewer write requests for placing orders. CQRS allows each operation to be handled efficiently.
- These patterns become especially valuable in systems where read and write operations have different performance characteristics with different latency or resource requirements.

5. Event Sourcing

- Think of Event Sourcing as keeping a journal of the live events. Instead of updating a record directly, we store events representing changes.
- This approach provides a complete history of the system and enables better auditing and debugging. Git Version control is a great example of event sourcing where each commits represents a change now with event sourcing.

6. Leader election

- Imagine a classroom of students electing a class representative in a distributor system.
- The leader election pattern ensures only one node is responsible for a specific task or resource. When the leader node fails, the remaining nodes elect a new leader.
- Use this pattern to manage distributed configurations. By having a designated leader, we can avoid conflicts and ensure consistent decision making across the distributed system.

7. Publisher/Subscriber

- Publisher/Subscriber pattern is like a newspaper delivery service. Publishers emit events without knowing who will receive them, and subscribers listen for events they're interested in.
- This pattern allows for better scalability and modularity.
- Complex applications pub/sub systems are well suited for scenarios where we need to propagate changes or updates across multiple components. For example, updating a user's profile across various services.

8. Sharding

- Sharding is like dividing a large pizza into smaller slices, making it easier to handle. It's a technique for distributing data across multiple nodes in a system.

- It improves performance and scalability. Each Shard contains a subset of the data, reducing the load on any single node.
- Databases like MongoDB and Cassandra use sharding to handle large amounts of data efficiently.
- Sharding can also help us achieve better data locality, reducing network latency and speeding up query execution.

9. Strangler Pattern

- This pattern is inspired by the Strangler fig tree, which grows around other trees and eventually replaces them. In software, the Strangler Pattern is a method for gradually replacing legacy systems with new implementations.
- Instead of performing a risky Big Bang migration, we can incrementally replace parts of this old system with new components.
- This approach can help us manage the risk and complexities associated with system migrations.

10. Load Balancing

- Distributes incoming network traffic across multiple servers to improve system performance, scalability, and availability.
- The goal is to prevent any single server from becoming overloaded while maintaining smooth and reliable service for users.

> *Don't forget to hit the Clap and Follow buttons to help me write more articles like this.*

References

Top 5 distributed system design patterns

Today, we explore 5 of the top distributed system design patterns you'll need to land your next senior back-end job.

www.educative.io

Distributed System Design Patterns

Key patterns referring to common design problems related to distributed systems:

medium.com

System Design Concepts | System Design Interview | Distributed Systems | Programming | Coding

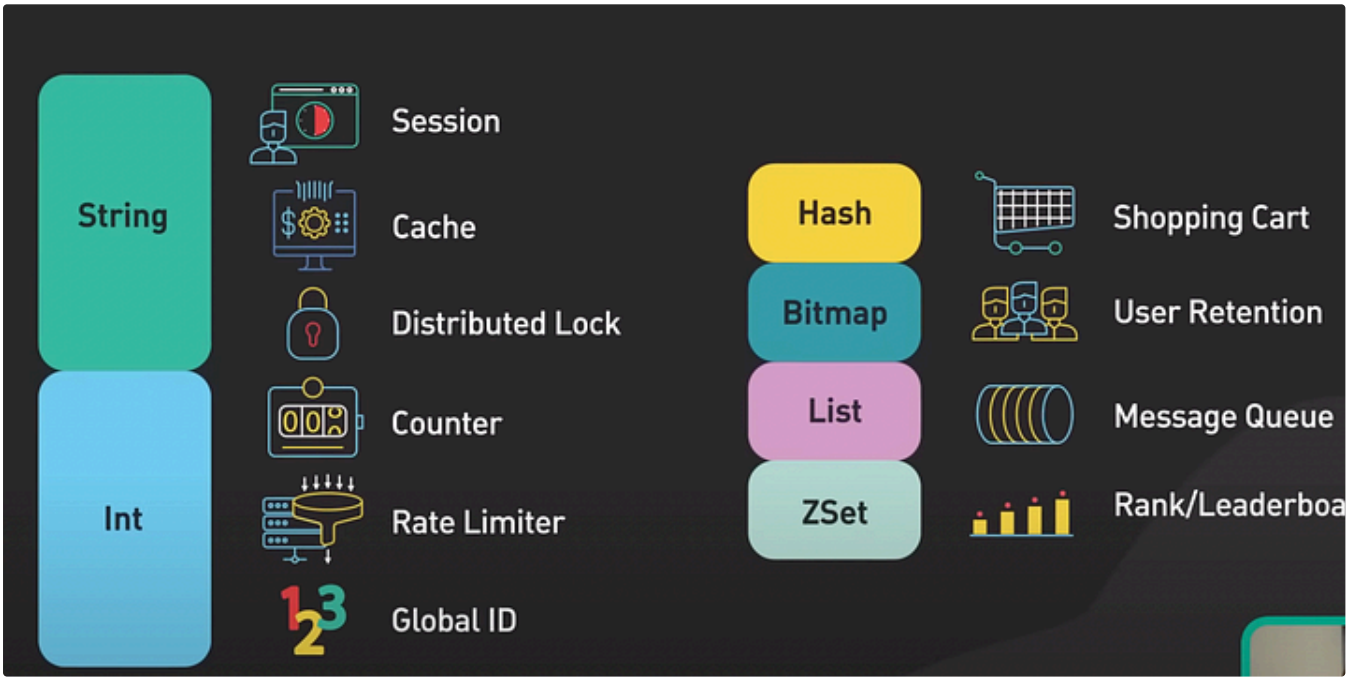
Follow

Written by Mahesh Saini

3.6K Followers

Enjoys coding, cycling, and swimming. Connect on LinkedIn: https://www.linkedin.com/in/mahesh-s-28529349/

More from Mahesh Saini



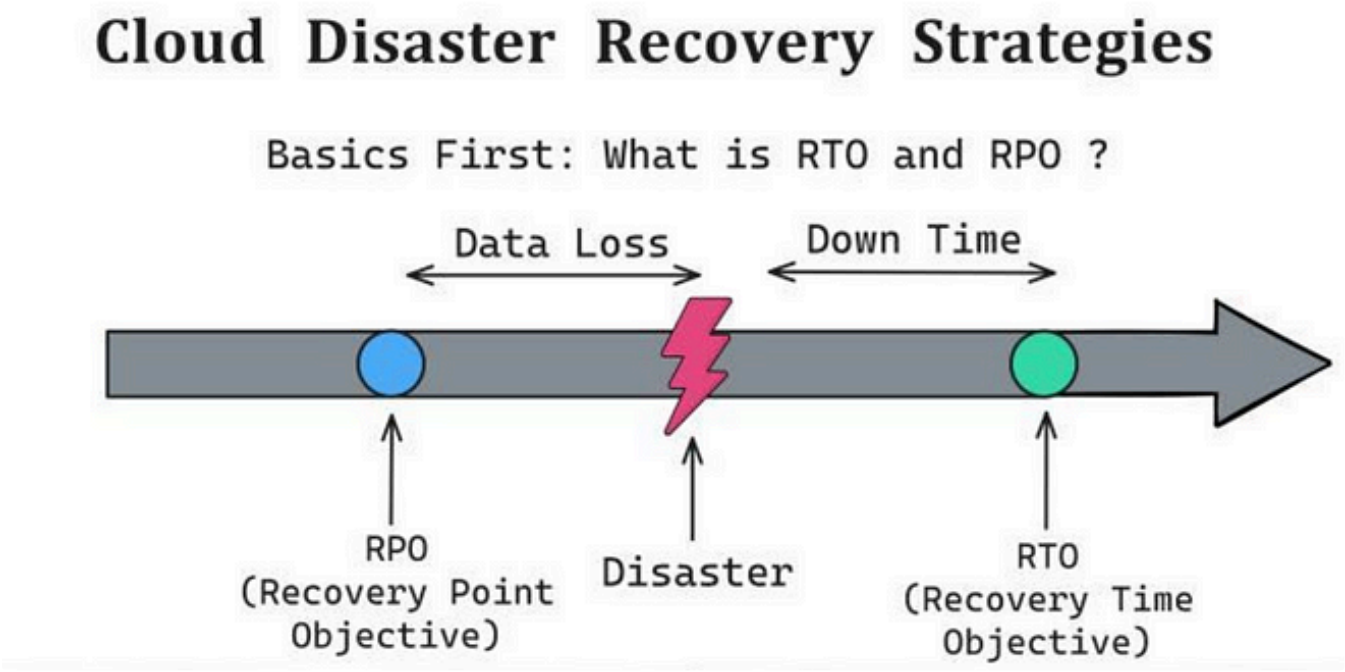
Mahesh Saini

Top 5 Redis Use Cases in Distributed Systems

We'll talk about the top use cases that have been better tested in production at various companies and at various scales, Get insights into...

· 5 min read · Jul 11, 2023

1.1K 10



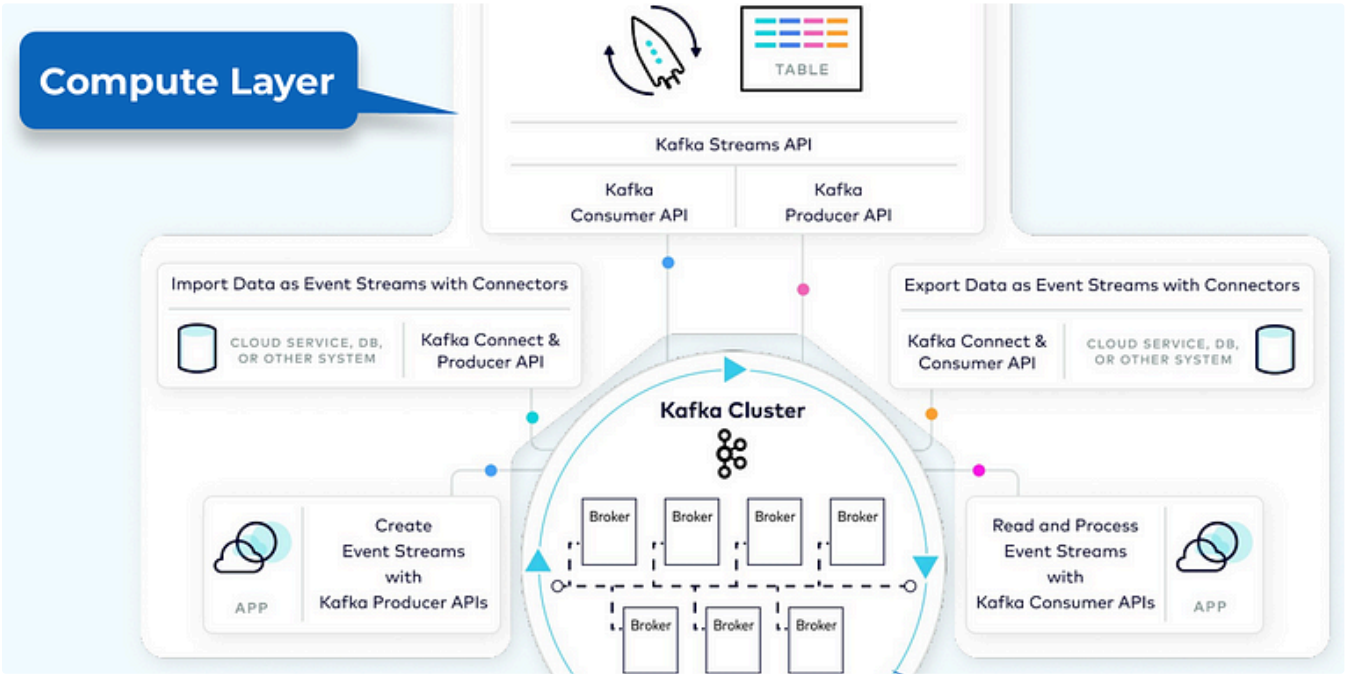
Mahesh Saini

System Design—4 Top Cloud Disaster Recovery Strategies

Cloud allows disaster recovery to be a very quick process, reducing the downtime that is suffered. It also provides the flexibility of...

4 min read · Sep 14, 2023

279 5



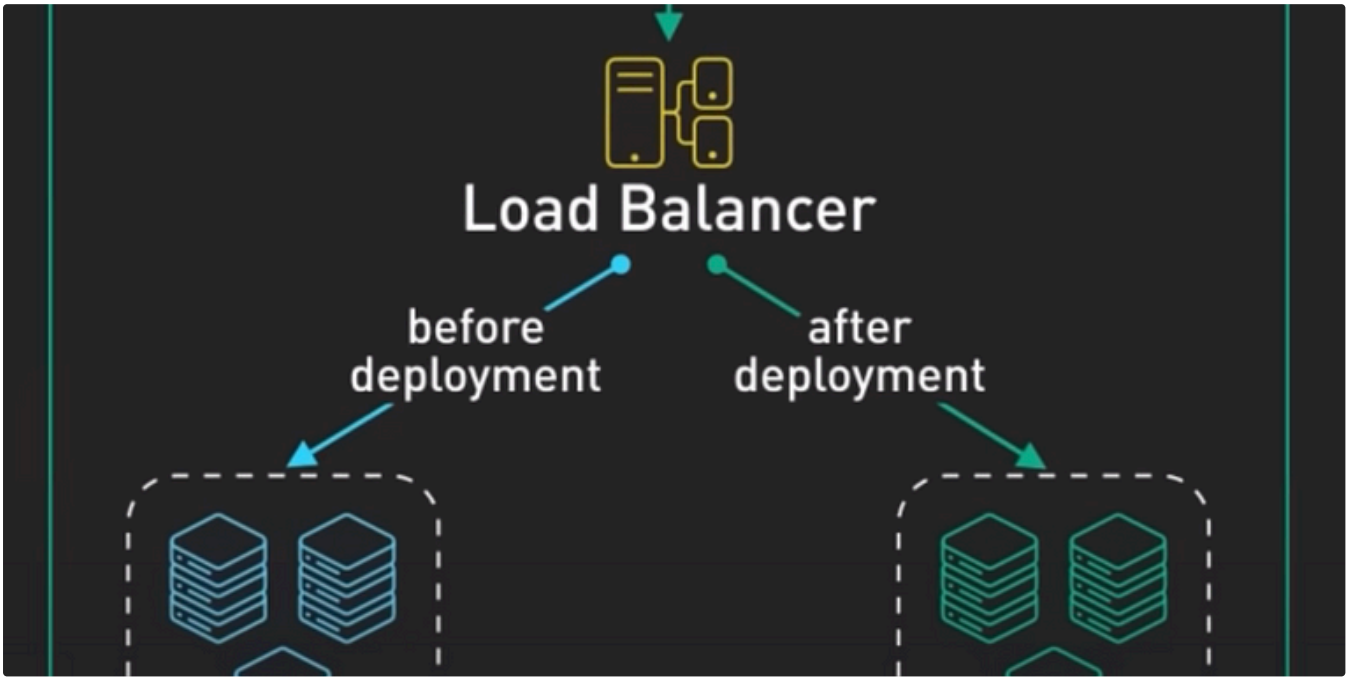
Mahesh Saini in The Life Titbits

Foundational Concepts of Kafka and Its Key Principles

Apache Kafka is a distributed event store and stream-processing platform. The project aims to provide a unified, high-throughput...

7 min read · May 2, 2023

246 1



Mahesh Saini

Top 5 Most-Used Deployment Strategies

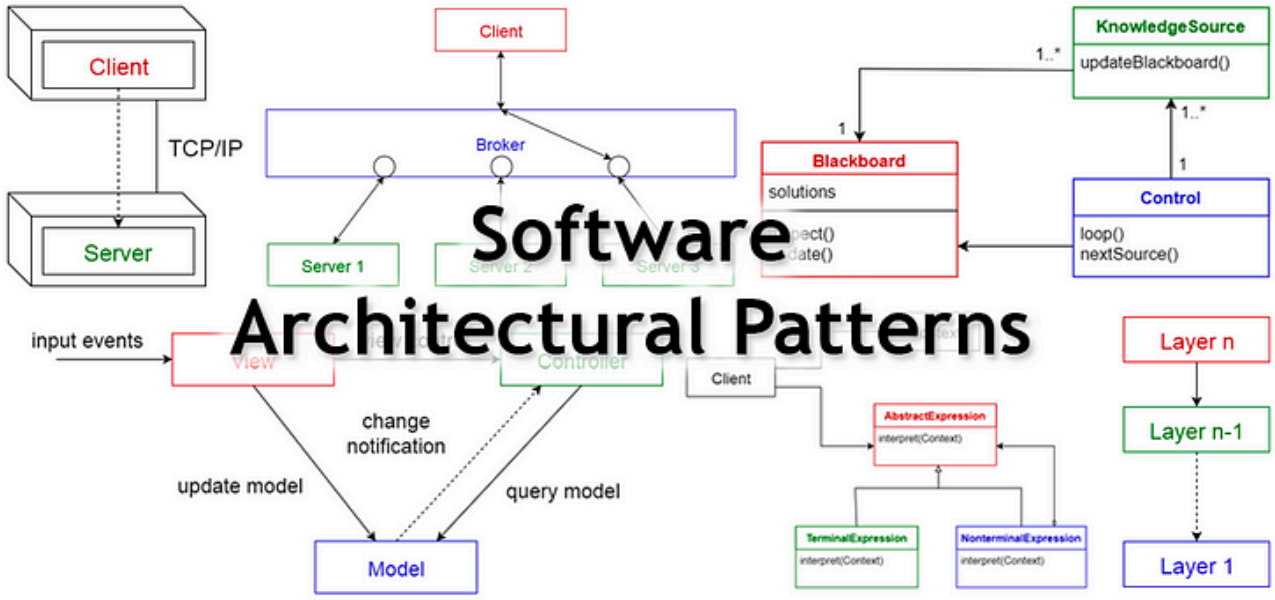
Deployment strategies define how you want to deliver your software. Organizations follow different deployment strategies based on their...

6 min read · Jul 4, 2023

578 6

See all from Mahesh Saini

Recommended from Medium



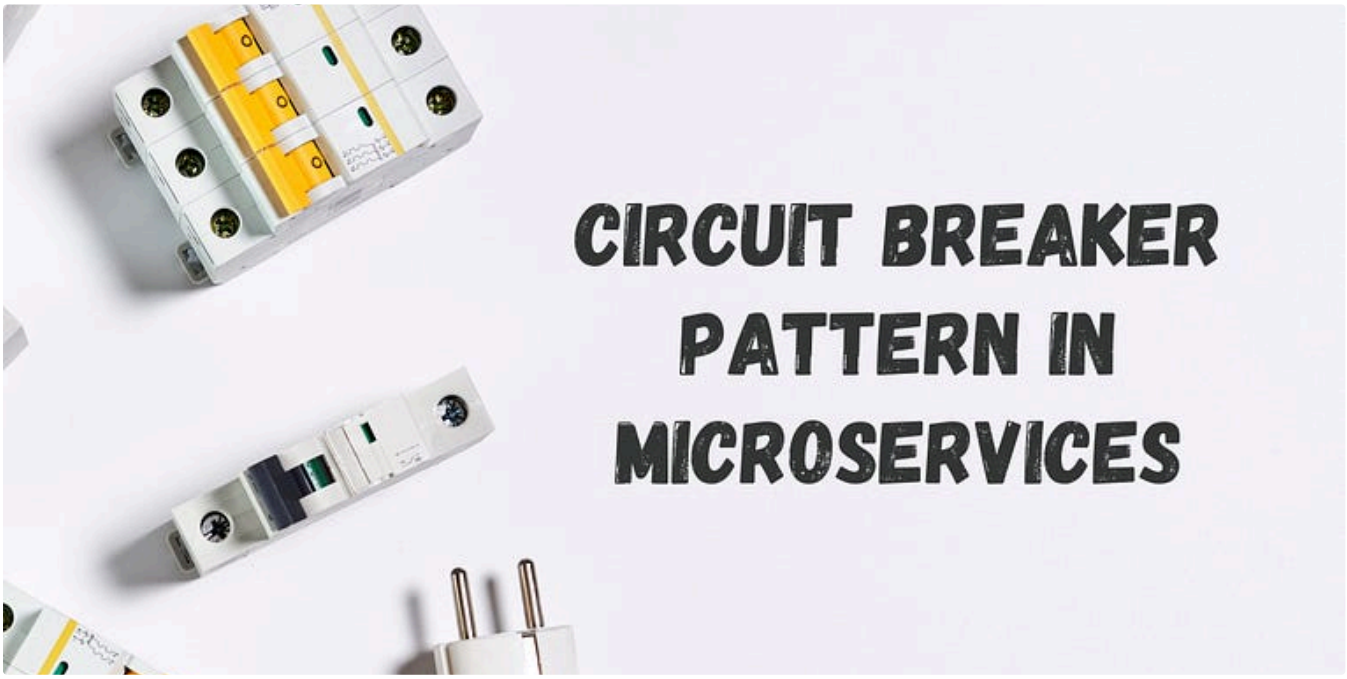
hubian

12 common software architecture styles, essential for architects

What is software architecture?

14 min read · Jan 8, 2024

53 1



Chameera Dulanga in Bits and Pieces

Circuit Breaker Pattern in Microservices

How to Use the Circuit Breaker Software Design Pattern to Build Microservices

6 min read · Jan 11, 2023

708 4

Lists

General Coding Knowledge

20 stories · 1270 saves

Stories to Help You Grow as a Software Developer

19 stories · 1100 saves

Coding & Development

11 stories · 639 saves

ChatGPT

21 stories · 663 saves

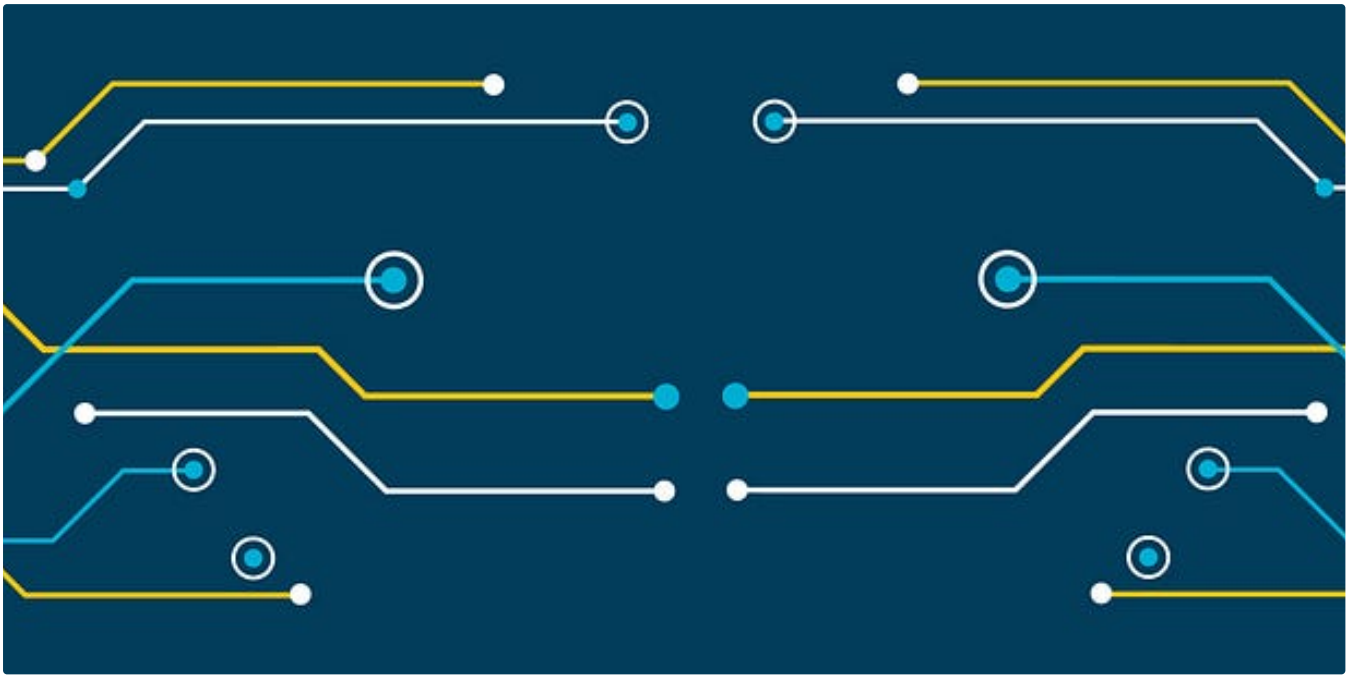
Crafting-Code

Top Microservices Interview Questions and Answers which You shouldn't miss

Microservices have become a crucial architectural approach. Whether you're a seasoned developer or a job seeker preparing for interviews...

5 min read · Jan 23, 2024

161 1



Capital One Tech in Capital One Tech

10 microservices design patterns for better architecture

Consider using these popular design patterns in your next microservices app and make organization more manageable.

11 min read · Jan 10, 2024

2.7K 16

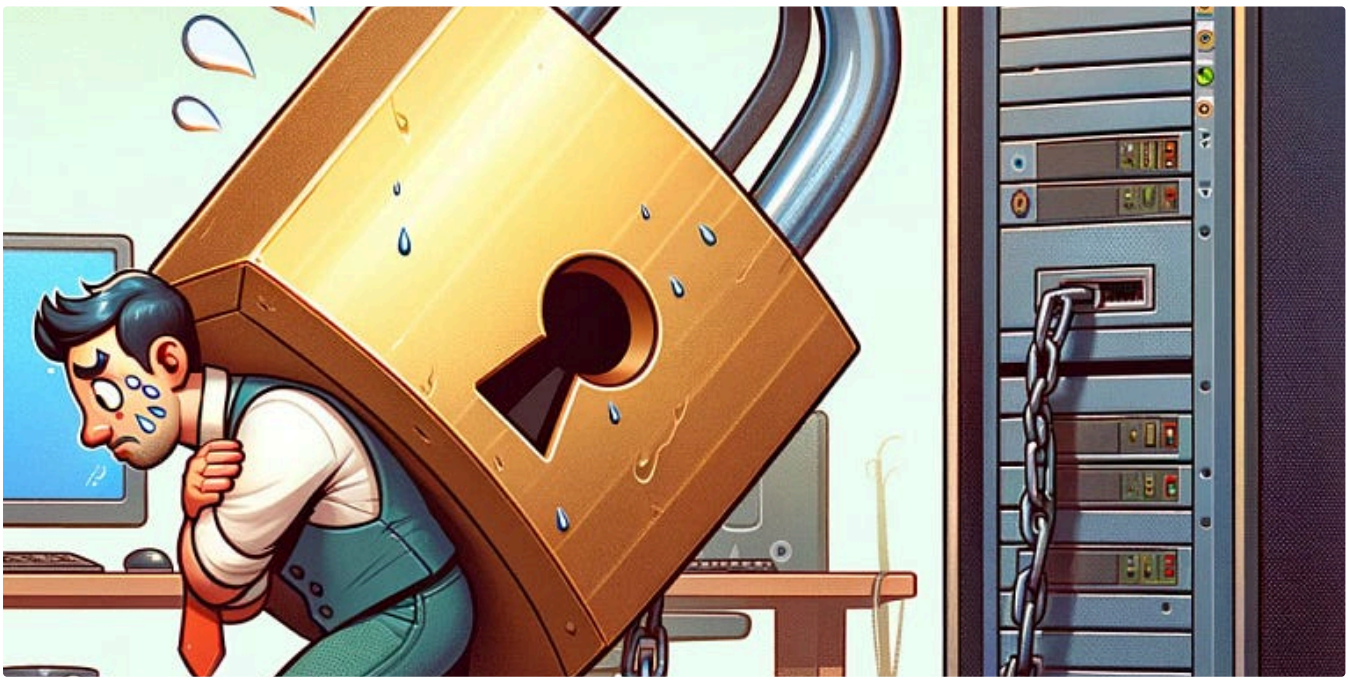
Talha Şahin

High-Level System Architecture of Booking.com

Take an in-depth look at the possible high-level architecture of Booking.com.

8 min read · Jan 10, 2024

5K 40



Mehmed Ali Çalışkan in Hexaworks-Papers

Managing User Authentication and Authorization in Microservices Architectures

In microservices architectures, user authentication and authorization play pivotal roles, often leveraging a range of technologies. This...

11 min read · Feb 1, 2024

 14  

See more recommendations