✅ 10 Real Interview Questions I Faced on Microservices – With My Answers  
  
Hey Connections👋,  
  
Over the past few months, I appeared for multiple interviews for Java Developer roles focused on Microservices architecture.  
  
Here are 10 real questions that I was asked in various interviews — with detailed answers I gave. Sharing to help others prepare better!  
  
🔹 1. What are Microservices?  
  
Microservices are an architectural style where applications are structured as a collection of loosely coupled, independently deployable services, each responsible for a specific business capability. Each service can be developed and deployed independently.  
  
🔹 2. How do Microservices communicate with each other?  
  
Mostly via REST APIs, gRPC, Message Brokers like RabbitMQ or Kafka. The communication can be synchronous (HTTP) or asynchronous (messaging).  
  
🔹 3. What is the role of Spring Cloud in Microservices?  
  
Spring Cloud provides tools for developers to quickly build common patterns like configuration management, service discovery, circuit breakers, routing, and more, which are essential for cloud-native microservice architectures.  
  
🔹 4. How do you handle service discovery?  
  
Using Eureka Server (Netflix OSS), or tools like Consul or Zookeeper, where services register themselves and can discover others without hardcoding URLs.  
  
🔹 5. What is an API Gateway and why is it used?  
  
API Gateway acts as a single entry point for all client requests. It helps in:  
  
Routing  
  
Authentication  
  
Rate limiting  
  
Logging  
Examples: Spring Cloud Gateway, Zuul, Kong, NGINX  
  
🔹 6. How do you implement centralized configuration?  
  
Using Spring Cloud Config Server, which stores all configuration files (e.g., in Git), allowing dynamic updates without redeployment.  
  
🔹 7. What is Circuit Breaker in Microservices?  
  
Circuit Breaker helps in fault tolerance. If a service is down, instead of failing the whole system, the circuit opens and returns a fallback response.  
Tools: Resilience4j, Hystrix (deprecated)  
  
🔹 8. What is Service Registry?  
  
It is a central place where services register themselves and discover others. Eureka Server is an example.  
  
🔹 9. How do you secure microservices?  
  
With OAuth2, JWT, and Spring Security. For API Gateway, use token validation. Internal communication may use mutual TLS (mTLS).  
  
🔹 10. What is the difference between Monolith and Microservices?  
  
Monolith: Single unit, tightly coupled

🟢Microservices Questions are must in any interview. if you are preparing for interview or want to level up microservices then below list will help you.   
  
Monolith vs. Microservices   
❓ What is the difference?   
✅ Monolith = One big app. Microservices = Many small apps working together.   
❓ When to use microservices?  
✅ When you need scalability, independent teams, and frequent updates.  
  
2. Designing a Microservice   
❓ How to start?   
✅ Identify one business function (e.g., "User Service"), define APIs, pick a database, and deploy separately.  
  
3. API Gateway  
❓ Why use it?   
✅ Single entry point for security, routing, and load balancing.  
  
4. REST vs. Messaging (Kafka/RabbitMQ)   
❓ When to use REST?   
✅ For immediate responses (e.g., fetching user data).   
❓ When to use messaging?   
✅ For slow/async tasks (e.g., sending emails).  
  
5. Circuit Breaker (Resilience4j)   
❓ What does it do?   
✅ Stops calling a failing service to avoid crashes (like a real circuit breaker).  
  
6. Load Balancing  
❓ Why needed?   
✅ Distributes traffic evenly across multiple service instances.  
  
7. Centralized Config (Spring Cloud Config)   
❓ Why useful?   
✅ Change configs (e.g., database URL) without redeploying services.  
  
8. Service Discovery (Eureka/Consul)   
❓ What does it do?   
✅ Helps services find each other dynamically (no hardcoded URLs).  
  
9. Feign Client vs. WebClient   
❓ Feign Client?   
✅ Simple REST calls (blocking).   
❓ WebClient?   
✅ Async, non-blocking calls (better for performance).  
  
10. Kafka for Events   
❓ Why use it?   
✅ Lets services communicate via events (e.g., "OrderPlaced").  
  
11. Database per Service   
❓ Why?   
✅ Prevents services from breaking each other’s data.  
  
12. Saga Pattern   
❓ What problem solves?   
✅ Manages transactions across services (e.g., "Pay Order → Update Inventory").  
  
13. API Gateway Security   
❓ How to secure?   
✅ Use HTTPS, rate limiting, and JWT validation.  
  
14. Observability   
❓ Why important?   
✅ Logs + metrics + tracing = Find bugs faster.  
  
15. Prometheus + Grafana  
❓ Prometheus?  
✅ Collects metrics (e.g., "CPU usage").   
❓ Grafana?   
✅ Shows metrics in dashboards.