Java/Spring Boot Production Debugging Interview Q&A;

1. API slow only in production

Check API gateway, load balancer, DB queries, thread pools, connection pools, GC, network latency, external APIs, logging overhead. Use APM to locate bottlenecks. Prefer pagination and optimized SQL.

2. Kafka lag increasing

Consumers are slower than producers due to slow processing, downstream APIs, partition imbalance, or insufficient concurrency. Increase partitions/concurrency and optimize processing.

3. Database connections exhausted

Look for connection leaks, long transactions, too-small HikariCP pool, or excessive concurrent requests. Always close connections or use @Transactional.

4. Autoscaling but response time increases

Scaling pods doesn't help if DB, Redis, Kafka, or external services are the bottleneck. Identify the real bottleneck before scaling.

5. Retry creates duplicate payments

Implement idempotency using an Idempotency-Key and a unique database constraint so retries don't create duplicate transactions.

6. Scheduled job runs multiple times

Each pod executes @Scheduled. Use distributed locking (e.g., ShedLock) or Quartz cluster mode.

7. Slow downstream affects platform

Use timeouts, circuit breakers, bulkheads, retries, and fallbacks so one dependency cannot block the whole system.

8. Random 500 errors

Investigate logs, stack traces, distributed tracing, serialization issues, race conditions, and intermittent NPEs.

9. Health checks pass but users fail

Basic health checks may only verify connectivity. Use readiness probes and synthetic end-to-end monitoring.

10. Cache returns stale data

Use TTL, @CacheEvict, cache invalidation strategies, or write-through/write-behind patterns.

11. Logs everywhere but debugging is hard

Use correlation IDs, Trace IDs, and distributed tracing (OpenTelemetry, Jaeger, Zipkin).

12. JVM memory increases

Investigate memory leaks using heap dumps. Common causes include static collections and ThreadLocal leaks.

13. Works in staging but fails behind gateway

Check authentication, JWT propagation, headers, CORS, TLS termination, and path rewriting.

14. Thread pools exhausted with low CPU

Blocking I/O causes threads to wait while CPU stays idle. Use async processing or non-blocking I/O.

15. Circuit breakers but cascading failures

Circuit breakers alone are insufficient. Combine timeout -> bulkhead -> retry -> circuit breaker -> fallback.