Microservices Fundamentals

1. What is Microservices architecture?
2. How is Microservices architecture different from Monolithic architecture?
3. What are the key characteristics of Microservices?
4. What problems does Microservices solve?
5. What are the disadvantages of Microservices?
6. When should you NOT use Microservices?
7. What is bounded context in Microservices?
8. What is loose coupling and why is it important?
9. What is high cohesion in Microservices?
10. What does “independent deployability” mean?
11. What is domain-driven design (DDD)?
12. How does DDD relate to Microservices?
13. What is service autonomy?
14. What is service granularity?
15. What is a single responsibility principle in Microservices?
16. How are Microservices deployed?
17. What is a stateless service?
18. Why should Microservices be stateless?
19. What is service ownership?
20. How does Microservices improve scalability?

Communication & Integration

21. How do Microservices communicate with each other?
22. Difference between synchronous and asynchronous communication?
23. REST vs gRPC – which is better and why?

24. What is message-based communication?
25. What is event-driven architecture?
26. What is API Gateway?
27. Why do we need an API Gateway?
28. What are the responsibilities of an API Gateway?
29. What is service-to-service communication?
30. What is request chaining?
31. What problems occur due to chatty communication?
32. How do you avoid tight coupling between services?
33. What is message broker?
34. Kafka vs RabbitMQ – differences?
35. What is schema evolution in messaging?

Data Management

36. How is data handled in Microservices?
37. Why should each Microservice have its own database?
38. Database-per-service pattern – explain.
39. How do you handle joins across Microservices?
40. What is eventual consistency?
41. What is strong consistency?
42. What is CQRS?
43. Difference between CQRS and CRUD?
44. What is event sourcing?
45. How does event sourcing work?
46. What is Saga pattern?

47. Choreography vs Orchestration Saga?
48. How do you handle distributed transactions?
49. What is two-phase commit and why is it avoided?
50. How do you ensure data integrity in Microservices?

Service Discovery & Configuration

51. What is service discovery?
52. Client-side vs server-side service discovery?
53. What is Eureka?
54. What is Consul?
55. What is DNS-based service discovery?
56. What is centralized configuration?
57. Why is config management important?
58. What is Spring Cloud Config?
59. How do you manage secrets in Microservices?
60. What is dynamic configuration refresh?

Resilience, Fault Tolerance & Performance

61. What is fault tolerance?
62. What is resilience in Microservices?
63. What is circuit breaker pattern?
64. How does circuit breaker work?
65. What is retry pattern?
66. What is bulkhead pattern?
67. What is timeout configuration?
68. What is fallback mechanism?
69. What is rate limiting?

70. What is throttling?
71. How do you prevent cascading failures?
72. What is load balancing?
73. Client-side vs server-side load balancing?
74. What is autoscaling?
75. How do you test resilience?

Security in Microservices

76. What are common security challenges in Microservices?
77. What is OAuth 2.0?
78. What is OpenID Connect?
79. JWT vs OAuth – differences?
80. How is authentication handled in Microservices?
81. How is authorization implemented?
82. What is service-to-service security?
83. What is mutual TLS (mTLS)?
84. What is Zero Trust Architecture?
85. How do you secure APIs?
86. What is API rate limiting for security?
87. How do you manage secrets securely?
88. What is token propagation?
89. What is API gateway security?
90. How do you handle PII data securely?

DevOps, CI/CD & Containers

91. How does Microservices impact DevOps?

92. What is CI/CD pipeline in Microservices?
93. Why are containers important?
94. Docker vs Virtual Machines?
95. What is Kubernetes?
96. What problems does Kubernetes solve?
97. What is a Pod?
98. What is a Service in Kubernetes?
99. What is Ingress?
100. What is Helm?
101. What is blue-green deployment?
102. What is canary deployment?
103. Rolling vs recreate deployment?
104. How do you rollback Microservices?
105. How do you version Microservices?

Observability & Monitoring

106. What is observability?
107. Difference between monitoring and observability?
108. What are the three pillars of observability?
109. What is distributed tracing?
110. What is correlation ID?
111. Tools for Microservices monitoring?
112. What is centralized logging?
113. What is Prometheus?
114. What is Grafana?
115. What is OpenTelemetry?

Design Patterns & Best Practices

116. What are common Microservices design patterns?

117. What is Strangler Fig pattern?

118. How do you decompose a monolith into Microservices?

119. What are common Microservices anti-patterns?

120. How do you decide Microservice boundaries?