1)What is a microservices architecture ?

- ✓ It is an architecture to develop complex applications.
- ✓ These built as a collection of small, independent services.
- ✓ These services communicate with each other through APIs.
- ✓ It will enable modular and decoupled systems.

2)What are the benefits of using a microservices architecture?

A. Scalability (Scale Up/Down and Scale In/out)

B. Resilience(Reduce Cascading Failures)(

C. Flexibility, faster development cycles.

D. Easier maintenance due to it's modular and independent nature.

3) Explain SOLID Principle?

- Single Responsibility Principle (SRP) : A class should have only one reason to change, meaning it should have a single responsibility or job.
- Open-Closed Principle(OCP) : Software entities should be open for extension but closed for modification.
- Liskov Substitution Principle(LSP) : Objects of a superclass should be able to be replaced with objects of its subclasses without breaking the behavior of the program.
- Interface Segregation Principle(ISP) : Creation of smaller , more cohesive interfaces instead of large ones that cover multiple scenarios.
- Dependency Inversion Principle(DIP) : This principle encourages the use of interfaces and dependency injection to decouple modules and make them more reusable and testable.

4)What is spring cloud and what are it's key features?

- ➢ Spring cloud is a framework built on top of spring boot.
- ➢ Used for development and deployment of microservices.
- ➢ Key features: service discovery and registration , load balancing , circuit breaking, distributed configuration, API gateway, and distributed tracing.

5)What is service discovery and how is it achieved in spring cloud?

- ✓ The process of dynamically locating and communicating with services
- ✓ In spring cloud, service discovery is achieved using Netflix Eureka or Consul.
- ✓ Allows services to register themselves and discover other services, enabling communication between microservices.

6)Explain the role of API gateways in microservices?

- ❖ Act as a single entry point for client requests to microservices.
- ❖ Handles dynamic routing, load balancing.
- ❖ The Gateway (old one is Zuul) library is used to implement AElPI gateways.
- ❖ Support integration with service discovery and authentication.

7)What is circuit breaking and how is it implemented using spring cloud?

- It is pattern that prevents cascading failures in microservices.
- Circuit breaking is implemented using the Netflix Hystrix library/Resilence4J.
- Monitors the health of dependent services and , if failures occur, opens the circuit to stop sending requests.
- It is ensuring the overall system stability

8)What is load balancing and how is it handled in microservices?

- Distribution of client requests across multiple instances of a services
- Spring cloud integrates with load balancers like Netflix Ribbon /Cloud Load balancer.
- These are client-side load balancing
- Handle load distribution without manual configuration

9)What are tools used to aggregate microservices log files?

- ELK(Elasticsearch , Logstash, Kibana) : Logstash is responsible for collecting and parsing log data. Elasticsearch indexes and stores the logs and kibana is a interface for log visualization and analysis.
- Splunk : Splunk is a powerful commercial tool that enables log aggregation , searching , monitoring and analysis. It offers features like real-time alerts, dashboards, and machine learning capabilities for log data.
- Fluentd : Fluentd is an open-source data collector that can aggregate and route log data from various sources to different destinations

10)What is distributed tracing and how is it implemented in spring cloud?

- It is a technique used to track and monitor requests as they flow through multiple microservices.
- It is implemented through integration with tracing systems : Zipkin
- Sleuth used for tracing information across microservices , transferred to Zipkin server using Zipkin client.

11) What is the purpose of spring cloud Config, and how does it work?

- It is centralized management of configuration properties for microservices.
- It uses configurations in a version-controlled repository and provides a configuration server.
- Microservices can retrieve their configuration information from the server at runtime, enabling dynamic and centralized configuration.

12)What are different types of spring cloud Config?

- ✓ Local File System : Configuration properties can be stored in a local file system. The configuration files are typically in yml or properties format.
- ✓ Git : Microservices can retrieve the configuration from the specified Git repository, allowing for version control and easy management.
- ✓ HashiCorp Vault: Configuration properties can be stored securely in Vault and the spring cloud Config server can retrieve them using appropriate authentication and authorization mechanisms.

13)What are the different approaches for inter-service communication in microservices?

- The different approaches for inter-service communication in microservice include synchronous communication through HTTP/REST APIs.
- Asynchronous messaging using message brokers like RabbitMQ or Apache Kafka.
- Event-driven communication using event buses or pub/sub mechanisms.

14)What is service orchestration and service choreography in microservices?

- Service orchestration is a centralized approach where a central component controls and coordinates the execution flow of microservices.

- In contrast, service choreography is a decentralized approach where microservices collaborate with each other directly, without a central controller.
- Service orchestration provides a more controlled and coordinated workflow , while service choreography offers greater flexibility to individual servies.

15)What is the role of containers and container orchestration platforms?

- ➢ It is lightweight and portable environment for packaging and deploying microservices.
- ➢ Consistency across different environments.
- ➢ Orchestration platforms , such as Kubernates and Docker Swarm, automate the management of containers at scale.
- ➢ They handle tasks like deployment ,scaling ,service discovery ,load balancing and fault tolerance in a distributed environment.

16)Explain the concept of event-driven architecture and how spring cloud supports it.

- ✓ Used for services communicate and react to events asynchronously.
- ✓ Spring cloud provides support for event-driven architecture through it's integration with messaging systems like RabbitMQ or Apache Kafka.
- ✓ Spring Cloud Stream and spring cloud Bus enable the implementation of event-driven petterns.
- ✓ Allowing services to publish and subscribe to events , facilitating loose coupling and scalability in the system

17)What are the challenges and considerations for testing microservices?

- Managing test data
- Orchestrating test environments ,ensuring proper isolation, handling dependencies
- Designing effective end-to- end tests.
- Selecting appropriate testing frameworks
- Comprehensive test coverage across the distributed system.

18)How can you handle authentication in microservices?

- ❖ Spring cloud provides various mechanisms to handle security and authentication in microservices.
- ❖ Integration with spring security , OAuth2 and JSON Web Tokens)JWT).
- ❖ These tools enable implementing authentication and authorization mechanisms , securing endpoints and managing user roles and permissions across microservies.

19)What is the role of centralized logging in microservices and how can it be achieved ?

- ✓ It helps collect and analyze logs form different services.
- ✓ Aiding in monitoring , troubleshooting and identifying issues across the distributed system.
- ✓ Spring cloud integrates with logging frameworks like ELK(Elasticsearch, Logstash, Kibana) or Splunk , allowing aggregation and analysis of logs from microservices in a centralized manner/

20)How does spring cloud handle service versioning and compatibility?

- Spring cloud does not provide a built-in mechanism for service versioning .
- It can be achieved through good API design practices such as using semantic versioning , backward compatibility and managing API contracts.
- Tools like spring cloud contract can help verify compatibility between service versions by providing consumer-driven contract tests.
- Additionally , using API gateways and service registries can assist in managing and routing requests to different versions of service based on their compatibility.