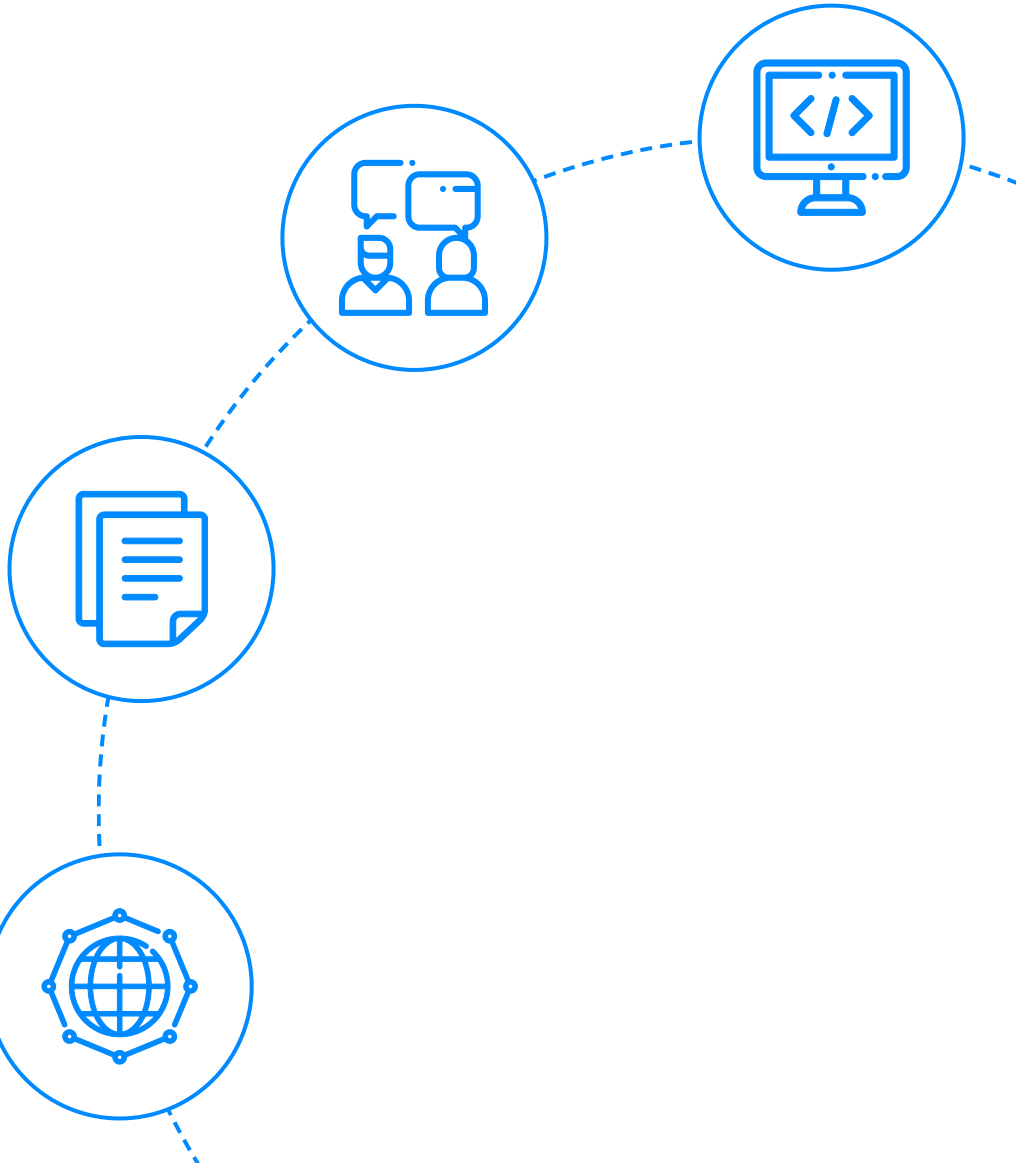
InterviewBit

Microservices Interview Questions



To view the live version of the page, click here.

Contents

Microservices Interview Questions for Freshers

Microservices Interview Questions for Experienced

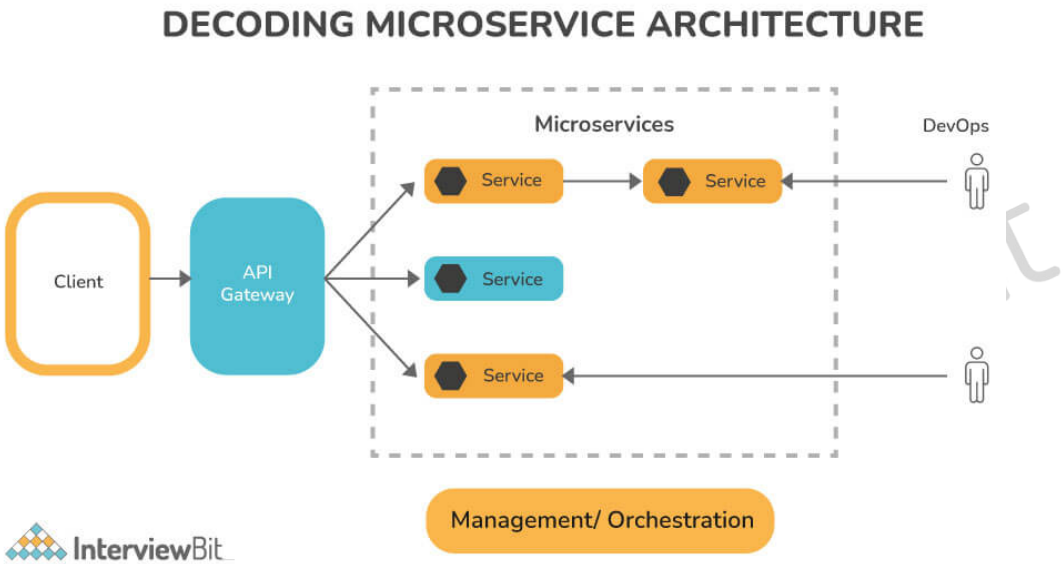
20. What do you mean by Semantic Monitoring?
21. Explain continuous monitoring.
22. What do you mean by Domain driven design?
23. Explain OAuth.
24. What do you mean by Distributed Transaction?
25. Explain Idempotence and its usage.
26. What do you mean by end-to-end microservices testing?
27. Explain the term Eureka in Microservices.
28. Explain the way to implement service discovery in microservices architecture.
29. Explain the importance of reports and dashboards in microservices.
30. What are Reactive Extensions in Microservices?
31. Explain type of tests mostly used in Microservices.
32. What do you mean by Mike Cohn's Test Pyramid?
33. Explain Container in Microservices.
34. What is the main role of docker in microservices?

Let's get Started

What do you mean by Microservice?

Microservices, also known as Microservices Architecture, is basically an SDLC approach in which large applications are built as a collection of small functional modules. It is one of the most widely adopted architectural concepts within software development. In addition to helping in easy maintenance, this architecture also makes development faster. Additionally, microservices are also a big asset for the latest methods of software development such as DevOps and Agile. Furthermore, it helps deliver large, complex applications promptly, frequently, and reliably. Applications are modeled as collections of services, which are:

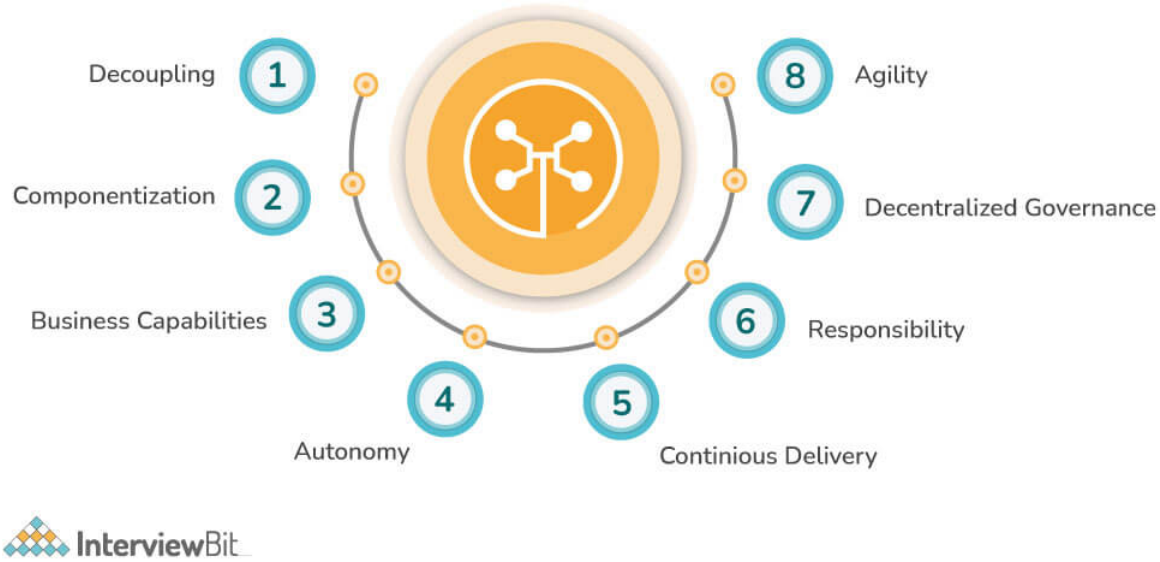
- Maintainable and testable
- Loosely coupled
- Independently deployable
- Designed or organized around business capabilities
- Managed by a small team



Microservices Interview Questions for Freshers

1. Write main features of Microservices.

Some of the main features of Microservices include:



- **Decoupling**: Within a system, services are largely decoupled. The application as a whole can therefore be easily constructed, altered, and scalable
- **Componentization**: Microservices are viewed as independent components that can easily be exchanged or upgraded
- **Business Capabilities**: Microservices are relatively simple and only focus on one service
- **Team autonomy**: Each developer works independently of each other, allowing for a faster project timeline
- **Continuous Delivery**: Enables frequent software releases through systematic automation of software development, testing, and approval
- **Responsibility:** Microservices are not focused on applications as projects. Rather, they see applications as products they are responsible for
- **Decentralized Governance:** Choosing the right tool according to the job is the goal. Developers can choose the best tools to solve their problems
- **Agility:** Microservices facilitate agile development. It is possible to create new features quickly and discard them again at any time.

2. Write main components of Microservices.

Some of the main components of microservices include:

- Containers, Clustering, and Orchestration
- IaC [Infrastructure as Code Conception]
- Cloud Infrastructure
- API Gateway
- Enterprise Service Bus
- Service Delivery

3. What are the benefits and drawbacks of Microservices?

Benefits:

- Self-contained, and independent deployment module.
- Independently managed services.
- In order to improve performance, the demand service can be deployed on multiple servers.
- It is easier to test and has fewer dependencies.
- A greater degree of scalability and agility.
- Simplicity in debugging & maintenance.
- Better communication between developers and business users.
- Development teams of a smaller size.

Drawbacks:

- Due to the complexity of the architecture, testing and monitoring are more difficult.
- Lacks the proper corporate culture for it to work.
- Pre-planning is essential.
- Complex development.
- Requires a cultural shift.
- Expensive compared to monoliths.
- Security implications.
- Maintaining the network is more difficult.

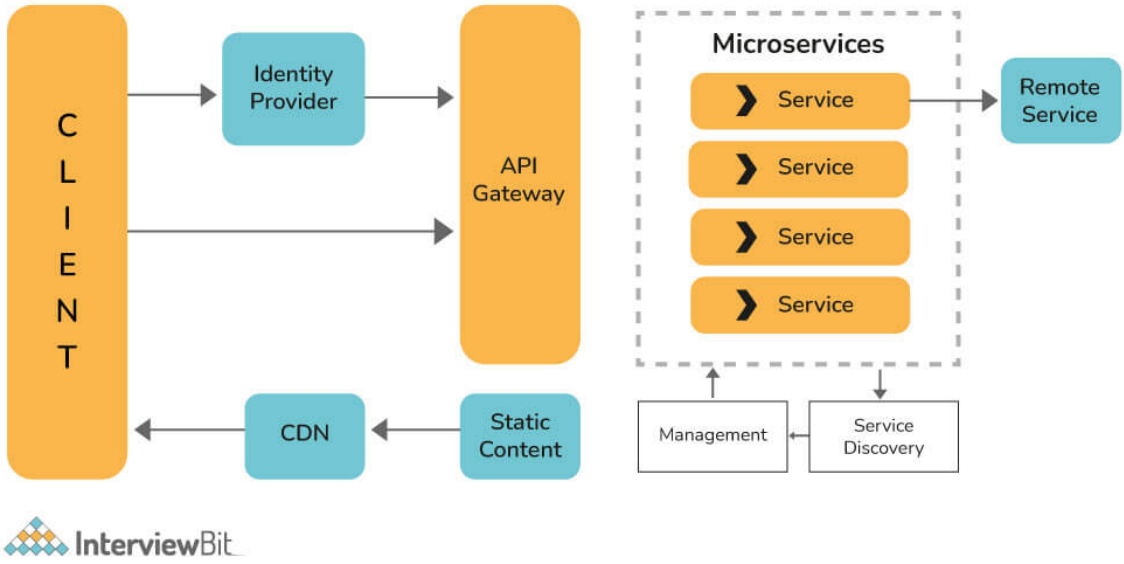
4. Name three common tools mostly used for microservices.

Three common tools used for microservices include:

- Wiremock
- Docker
- Hystrix

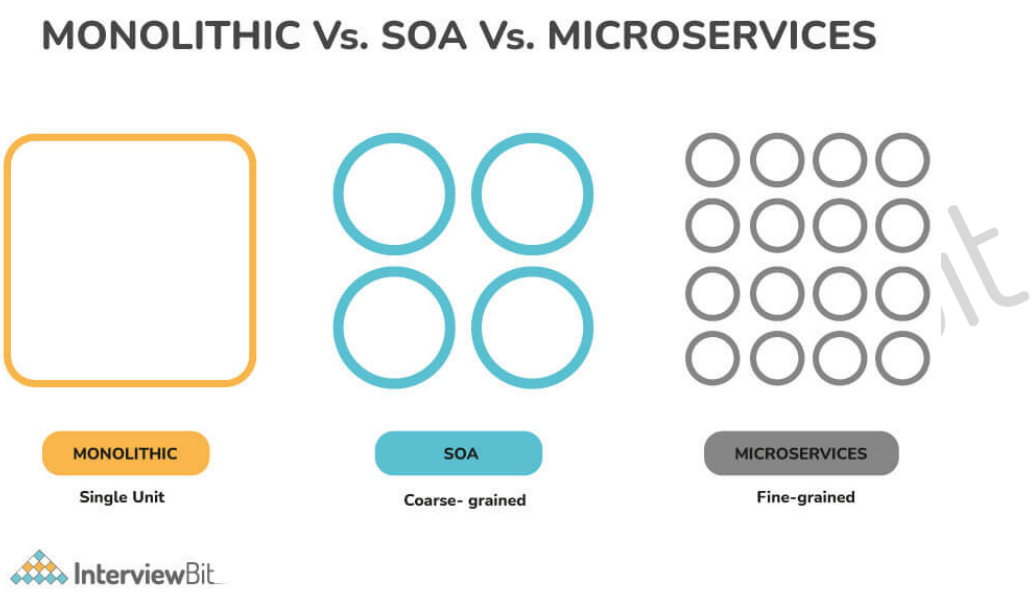
5. Explain the working of Microservice Architecture.

Microservice architectures consist of the following components:



- **Clients**: Different users send requests from various devices.
- **Identity Provider**: Validate a user's or client's identity and issue security tokens.
- **API Gateway:** Handles the requests from clients.
- **Static Content:** Contains all of the system's content.
- **Management:** Services are balanced on nodes and failures are identified.
- **Service Discovery:** A guide to discovering the routes of communication between microservices.
- **Content Delivery Network:** Includes distributed network of proxy servers and their data centers.
- **Remote Service:** Provides remote access to data or information that resides on networked computers and devices.

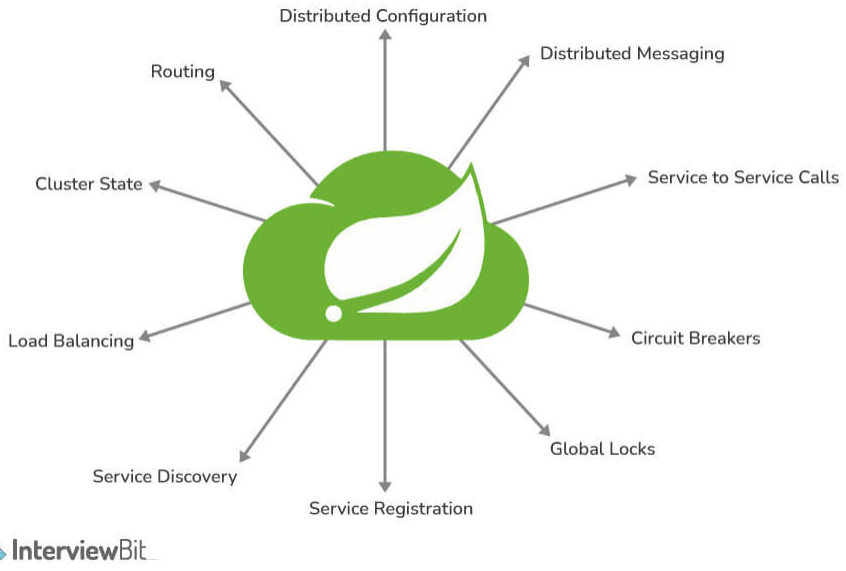
6. Write difference between Monolithic, SOA and Microservices Architecture.



- **Monolithic Architecture**: It is "like a big container" where all the software components of an application are bundled together tightly. It is usually built as one large system and is one code-base.
- **SOA (Service-Oriented Architecture)**: It is a group of services interacting or communicating with each other. Depending on the nature of the communication, it can be simple data exchange or it could involve several services coordinating some activity.
- **Microservice Architecture**: It involves structuring an application in the form of a cluster of small, autonomous services modeled around a business domain. The functional modules can be deployed independently, are scalable, are aimed at achieving specific business goals, and communicate with each other over standard protocols.

7. Explain spring cloud and spring boot.

Spring Cloud: In Microservices, the Spring cloud is a system that integrates with external systems. This is a short-lived framework designed to build applications quickly. It contributes significantly to microservice architecture due to its association with finite amounts of data processing. Some of the features of spring cloud are shown below:



Spring Boot: Spring Boot is an open-sourced, Java-based framework that provides its developers with a platform on which they can create stand-alone, production-grade Spring applications. In addition to reducing development time and increasing productivity, it is easily understood.

8. What is the role of actuator in spring boot?

A spring boot actuator is a project that provides restful web services to access the current state of an application that is running in production. In addition, you can monitor and manage application usage in a production environment without having to code or configure any of the applications.

9. Explain how you can override the default properties of Spring boot projects.

By specifying properties in the application.properties file, it is possible to override the default properties of a spring boot project.

Example:
In Spring MVC applications, you need to specify a suffix and prefix. You can do this by adding the properties listed below in the application.properties file.

- For suffix – spring.mvc.view.suffix: .jsp
- For prefix – spring.mvc.view.prefix: /WEB-INF/

10. What issues are generally solved by spring clouds?

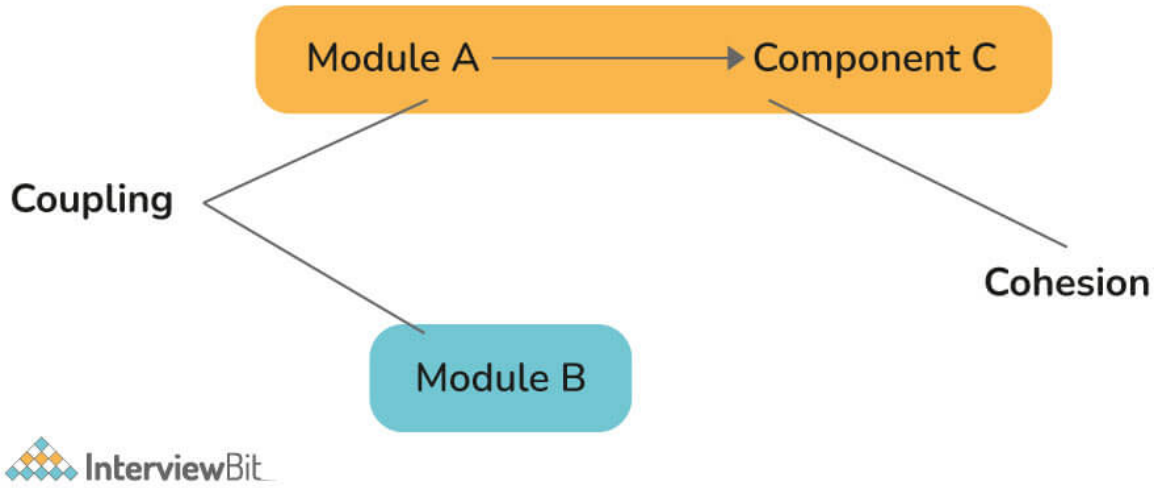
The following problems can be solved with spring cloud:

- **Complicated issues caused by distributed systems:** This includes network issues, latency problems, bandwidth problems, and security issues.
- **Service Discovery issues:** Service discovery allows processes and services to communicate and locate each other within a cluster.
- **Redundancy issues:** Distributed systems can often have redundancy issues.
- **Load balancing issues:** Optimize the distribution of workloads among multiple computing resources, including computer clusters, central processing units, and network links.
- **Reduces performance issues:** Reduces performance issues caused by various operational overheads.

11. What do you mean by Cohesion and Coupling?

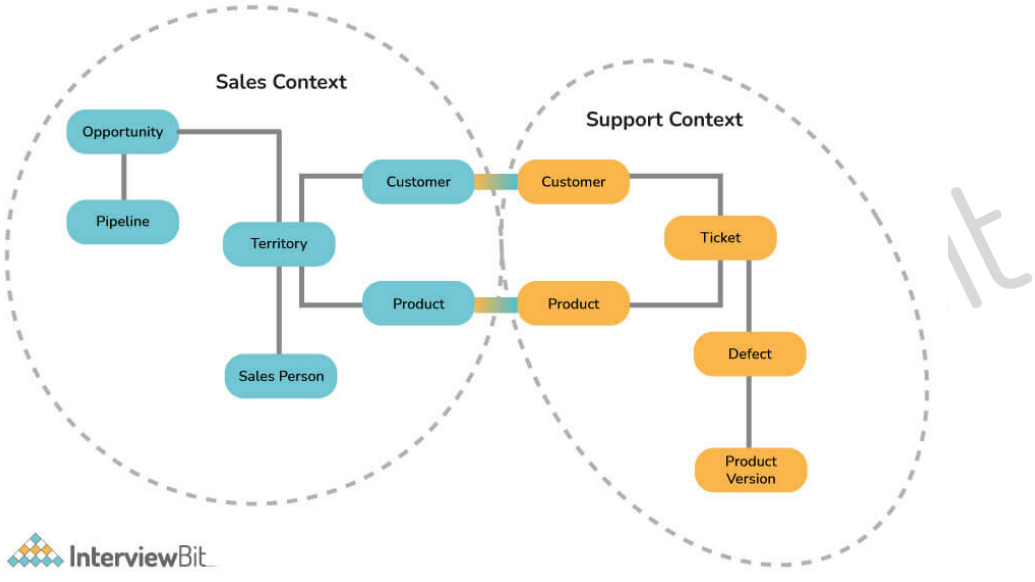
Coupling: It is defined as a relationship between software modules A and B, and how much one module depends or interacts with another one. Couplings fall into three major categories. Modules can be highly coupled (highly dependent), loosely coupled, and uncoupled from each other. The best kind of coupling is loose coupling, which is achieved through interfaces.

Cohesion: It is defined as a relationship between two or more parts/elements of a module that serves the same purpose. Generally, a module with high cohesion can perform a specific function efficiently without needing communication with any other modules. High cohesion enhances the functionality of the module.



12. What do you mean by Bounded Context?

A Bounded Context is a central pattern in DDD (Domain-Driven Design), which deals with collaboration across large models and teams. DDD breaks large models down into multiple contexts to make them more manageable. Additionally, it explains their relationship explicitly. The concept promotes an object-oriented approach to developing services bound to a data model and is also responsible for ensuring the integrity and mutability of said data model.



13. Write the fundamental characteristics of Microservice Design.

- **Based on Business Capabilities**: Services are divided and organized around business capabilities.
- **Products not projects:** A product should belong to the team that handles it.
- **Essential messaging frameworks:** Rely on functional messaging frameworks: Eliminate centralized service buses by embracing the concept of decentralization.
- **Decentralized Governance:** The development teams are accountable for all aspects of the software they produce.
- **Decentralized data management:** Microservices allow each service to manage its data separately.
- **Automated infrastructure:** These systems are complete and can be deployed independently.
- **Design for failure:** Increase the tolerance for failure of services by focusing on continuous monitoring of the applications.

14. What are the challenges that one has to face while using Microservices?

The challenges that one has to face while using microservices can be both functional and technical as given below:

Functional Challenges:

- Require heavy infrastructure setup.
- Need Heavy investment.
- Require excessive planning to handle or manage operations overhead.

Technical Challenges:

- Microservices are always interdependent. Therefore, they must communicate with each other.
- It is a heavily involved model because it is a distributed system.
- You need to be prepared for operations overhead if you are using Microservice architecture.
- To support heterogeneously distributed microservices, you need skilled professionals.
- It is difficult to automate because of the number of smaller components. For that reason, each component must be built, deployed, and monitored separately.
- It is difficult to manage configurations across different environments for all components.
- Challenges associated with deployment, debugging, and testing.

15. Explain PACT in microservices.

PACT is defined as an open-source tool that allows service providers and consumers to test interactions in isolation against contracts that have been made to increase the reliability of microservice integration. It also offers support for numerous languages, such as Ruby, Java, Scala, .NET, JavaScript, Swift/Objective-C.

16. Explain how independent microservices communicate with each other.

Communication between microservices can take place through:

- HTTP/REST with JSON or binary protocol for request-response
- Websockets for streaming.
- A broker or server program that uses advanced routing algorithms.

RabbitMQ, Nats, Kafka, etc., can be used as message brokers; each is built to handle a particular message semantic. You can also use Backend as a Service like Space Cloud to automate your entire backend.

17. What do you mean by client certificates?

The client certificate is a type of digital certificate that generally allows client systems to authenticate their requests to remote servers. In many mutual authentication designs, it plays a key role in providing strong assurance of the requestor's identity.

18. Explain CDC.

As the name implies, CDC (Consumer-Driven Contract) basically ensures service communication compatibility by establishing an agreement between consumers and service providers regarding the format of the data exchanged between them. An agreement like this is called a contract. Basically, it is a pattern used to develop Microservices so that they can be efficiently used by external systems.

19. Name some famous companies that use Microservice architecture.

Microservices architecture has replaced monolithic architecture for most large-scale websites like:

- Twitter
- Netflix
- Amazon, etc.

Microservices Interview Questions for Experienced

The semantic monitoring method, also called synthetic monitoring, uses automated tests and monitoring of the application to identify errors in business processes. This technology provides a deeper look into the transaction performance, service availability, and overall application performance to identify performance issues of microservices, catch bugs in transactions and provide an overall higher level of performance.

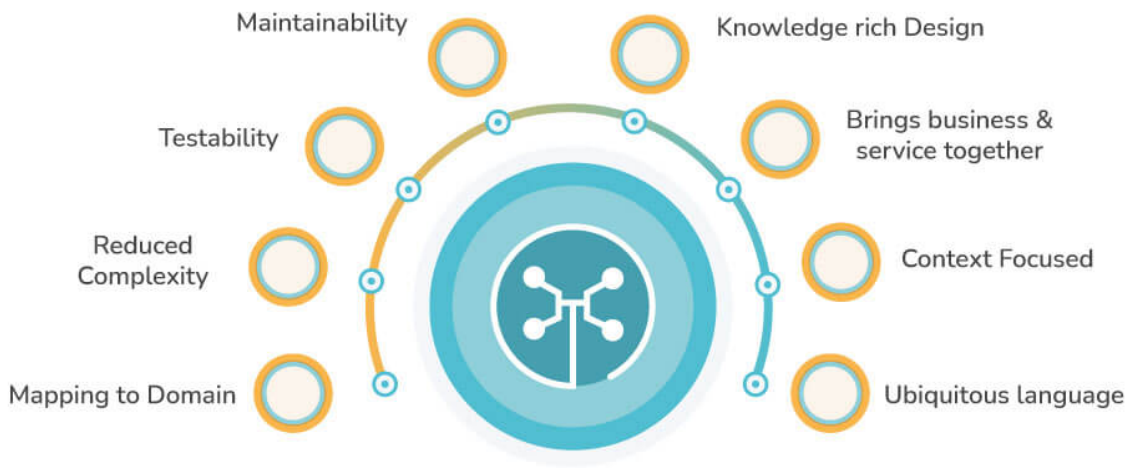
21. Explain continuous monitoring.

Continuous monitoring involves identifying compliance and risk issues in a company's financial and operational environment. It consists of people, processes, and working systems that support efficient and effective operations.

22. What do you mean by Domain driven design?

DDD (Domain-Driven-Design) is basically an architectural style that is based on Object-Oriented Analysis Design approaches and principles. In this approach, the business domain is modeled carefully in software, without regard to how the system actually works. By interconnecting related components of the software system into a continuously evolving system, it facilitates the development of complex systems. There are three fundamental principles underlying it as shown below:

- Concentrate on the core domain and domain logic.
- Analyze domain models to find complex designs.
- Engage in regular collaboration with the domain experts to improve the application model and address emerging domain issues.



23. Explain OAuth.

Generally speaking, OAuth (Open Authorization Protocol) enables users to authenticate themselves with third-party service providers. With this protocol, you can access client applications on HTTP for third-party providers such as GitHub, Facebook, etc. Using it, you can also share resources on one site with another site without requiring their credentials.

24. What do you mean by Distributed Transaction?

Distributed transactions are an outdated approach in today's microservice architecture that leaves the developer with severe scalability issues. Transactions are distributed to several services that are called to complete the transaction in sequence. With so many moving parts, it is very complex and prone to failure.

25. Explain Idempotence and its usage.

The term 'idempotence' refers to the repeated performance of a task despite the same outcome. In other words, it is a situation in which a task is performed repeatedly with the end result remaining the same.

Usage: When the remote service or data source receives instructions more than once, Idempotence ensures that it will process each request once.

26. What do you mean by end-to-end microservices testing?

Usually, end-to-end (E2E) microservice testing is an uncoordinated, high-cost technique that is used to ensure that all components work together for a complete user journey. Usually, it is done through the user interface, mimicking how it appears to the user. It also ensures all processes in the workflow are working properly.

27. Explain the term Eureka in Microservices.

Eureka Server, also referred to as Netflix Service Discovery Server, is an application that keeps track of all client-service applications. As every Microservice registers to Eureka Server, Eureka Server knows all the client applications running on the different ports and IP addresses. It generally uses Spring Cloud and is not heavy on the application development process.

28. Explain the way to implement service discovery in microservices architecture.

There are many ways to set up service discovery, but Netflix's Eureka is the most efficient. This is a hassle-free procedure that doesn't add much weight to the application. It also supports a wide range of web applications. A number of annotations are provided by Spring Cloud to make its use as simple as possible and to hide complex concepts.

29. Explain the importance of reports and dashboards in microservices.

Monitoring a system usually involves the use of reports and dashboards. Using reports and dashboards for microservices can help you:

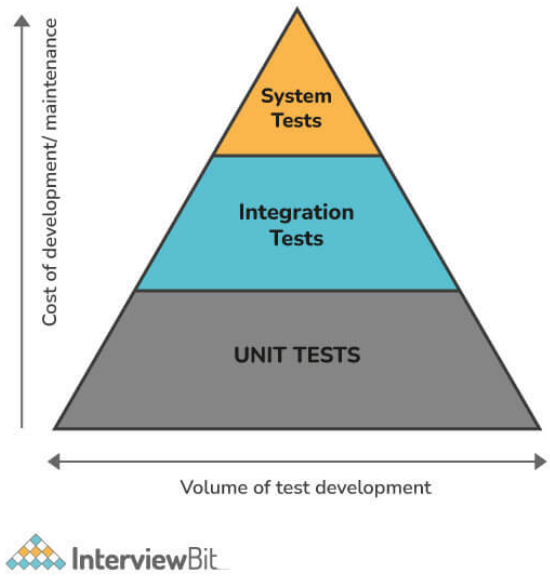
- Determine which microservices support which resources.
- Determine which services are impacted whenever changes are made or occur to components.
- Make documentation easy to access whenever needed.
- Review deployed component versions.
- Determine the level of maturity and compliance from the components.

30. What are Reactive Extensions in Microservices?

A reactive extension, also known as Rx, is basically a design approach that calls multiple services and then generates a single response by combining the results. The calls can either be blocking or not blocking, synchronous or asynchronous. A popular tool in distributed systems, Rx works exactly opposite to legacy flows.

31. Explain type of tests mostly used in Microservices.

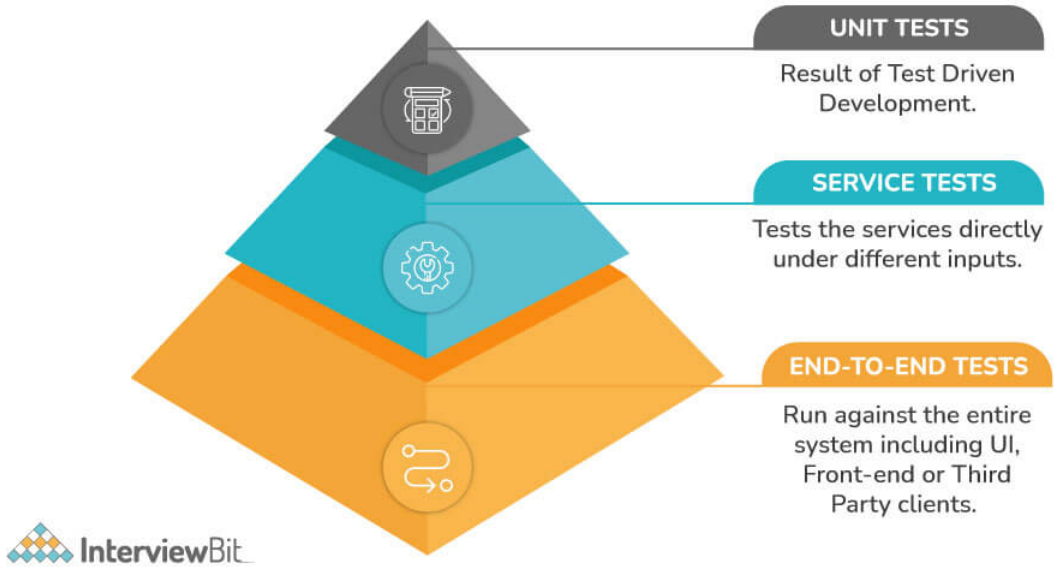
As there are multiple microservices working together, microservice testing becomes quite complex when working with microservices. Consequently, tests are categorized according to their level:



Botton-level tests: The bottom-level tests are those that deal with technology, such as unit tests and performance tests. This is a completely automated process.
Middle-level tests: In the middle, we have exploratory tests such as stress tests and usability tests.
Top-level tests: In the top-level testing, we have a limited number of acceptance tests. The acceptance tests help stakeholders understand and verify the software features.

32. What do you mean by Mike Cohn's Test Pyramid?

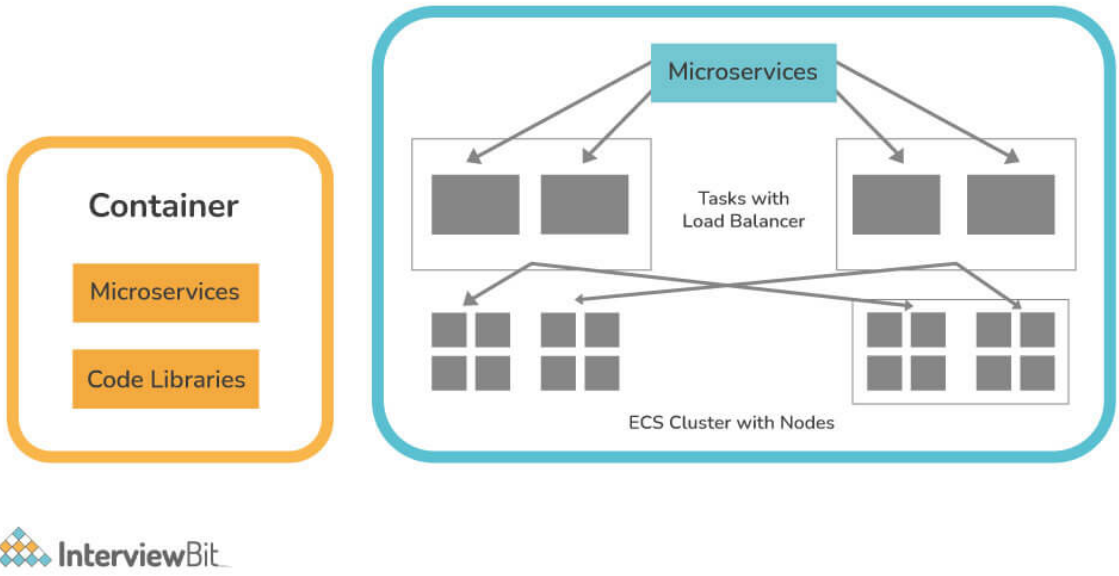
Mike Cohn's Test Pyramid explains the different types of automated tests needed for software development. The test pyramid is basically used to maximize automation at all levels of testing, including unit testing, service level testing, UI testing, etc. The pyramid also states that unit tests are faster and more isolated, while UI tests, which are at the top, are more time-consuming and are centered around integration.



In accordance with the pyramid, the number of tests should be highest at the first layer. At the service layer, fewer tests should be performed than at the unit test level, but greater than that at the end-to-end level.

33. Explain Container in Microservices.

Containers are useful technologies for allocating and sharing resources. It is considered the most effective and easiest method for managing microservice-based applications to develop and deploy them individually. Using Docker, you may also encapsulate a microservice along with its dependencies in a container image, which can then be used to roll on-demand instances of the microservice without any additional work.



34. What is the main role of docker in microservices?

Docker generally provides a container environment, in which any application can be hosted. This is accomplished by tightly packaging both the application and the dependencies required to support it. These packaged products are referred to as Containers, and since Docker is used to doing that, they are called Docker containers. Docker, in essence, allows you to containerize your microservices and manage these microservices more easily.

Conclusion:

Microservices architecture is a method of developing a large-scale application as a collection of small autonomous services developed for a business domain. Since its debut in 2011, microservices have become a popular technology, especially among organizations building forward-thinking applications. This list of Microservices interview questions was carefully constructed to assist the development community in their interviews. Hope these Microservices Architect Interview Questions would be helpful for your interview.

Links to More Interview Questions

C Interview Questions	Php Interview Questions	C Sharp Interview Questions
Web Api Interview Questions	Hibernate Interview Questions	Node Js Interview Questions
Cpp Interview Questions	Oops Interview Questions	Devops Interview Questions
Machine Learning Interview Questions	Docker Interview Questions	Mysql Interview Questions
Css Interview Questions	Laravel Interview Questions	Asp Net Interview Questions
Django Interview Questions	Dot Net Interview Questions	Kubernetes Interview Questions
Operating System Interview Questions	React Native Interview Questions	Aws Interview Questions
Git Interview Questions	Java 8 Interview Questions	Mongodb Interview Questions
Dbms Interview Questions	Spring Boot Interview Questions	Power Bi Interview Questions
Pl Sql Interview Questions	Tableau Interview Questions	Linux Interview Questions
Ansible Interview Questions	Java Interview Questions	Jenkins Interview Questions