**Q: 12 features to have to be called as microservices**

To qualify as a **microservices architecture**, a system should embody certain core principles and features. Here are **12 essential features** that characterize a microservices-based system:

**✅ 1. Independent Deployment**

Each microservice can be deployed independently without impacting others, allowing faster and safer rollouts.

**✅ 2. Single Responsibility (Business Capability)**

Each microservice is designed around a specific business function (e.g., Order Service, Payment Service), following the **Single Responsibility Principle**.

**✅ 3. Loose Coupling**

Services are **minimally dependent** on each other. Communication is typically done via APIs (e.g., REST, gRPC, messaging).

**✅ 4. High Cohesion**

Each service contains all necessary logic and data (within its domain), promoting maintainability and clarity.

**✅ 5. Technology Diversity**

Each service can be developed in different programming languages or frameworks, best suited to its needs (polyglot architecture).

**✅ 6. Decentralized Data Management**

Each microservice **owns its own database**, avoiding shared databases and ensuring data encapsulation.

**✅ 7. Resilience and Fault Tolerance**

Microservices handle failures gracefully (using **circuit breakers**, retries, fallbacks). A failure in one service should not cascade to others.

**✅ 8. Auto-Scaling**

Services can scale **independently** based on their own load, optimizing resource utilization.

**✅ 9. Service Discovery**

Microservices register themselves and discover others dynamically using tools like **Eureka**, **Consul**, or **Zookeeper**.

**✅ 10. API Gateway**

A unified entry point (e.g., **Spring Cloud Gateway**, **Kong**) manages routing, authentication, and load balancing.

**✅ 11. Observability**

Includes logging, metrics, and tracing for each service (using tools like **ELK**, **Prometheus**, **Grafana**, **Zipkin**, **Jaeger**).

**✅ 12. CI/CD and DevOps Enablement**

Automated pipelines enable frequent integration and delivery, ensuring faster feedback and reduced time to market.

**Bonus (Optional but Recommended):**

* **Security at the service level** (e.g., OAuth2, JWT)
* **Asynchronous Communication** with **message brokers** like Kafka or RabbitMQ
* **Domain-Driven Design (DDD)** principles