# CURSO de
# SPRING BOOT Avanzado y
# Microservicios con SPRING CLOUD

*(Patrones, seguridad, tolerancia a fallos y despliegue profesional)*

**OBJETIVO**

Formarte para ***crear y administrar arquitecturas de microservicios modernas con Spring Cloud***, dominando patrones clave, comunicación entre servicios, seguridad, tolerancia a fallos y despliegue automatizado, listo para entornos reales de alta demanda.

1. Arquitectura de Microservicios Profunda
   - 1.1. Patrones clave:
     - 1.1.1. API Gateway
     - 1.1.2. Service Discovery
     - 1.1.3. Circuit Breaker
     - 1.1.4. Config Server
     - 1.1.5. Event-Driven Architecture
   - 1.2. Comunicación entre servicios:
     - 1.2.1. Sincrónica (REST, Feign, gRPC)
     - 1.2.2. Asíncrona (RabbitMQ, Kafka)
   - 1.3. Estrategias de base de datos:
     - 1.3.1. Database-per-service
     - 1.3.2. Eventual consistency
     - 1.3.3. Outbox pattern

2. Spring Cloud Config Server
   - 2.1. Configuración centralizada con Git
   - 2.2. Configuración por entorno (dev, test, prod)
   - 2.3. Auto-reload con Spring Cloud Bus (RabbitMQ o Kafka)
   - 2.4. Seguridad en la configuración (encriptación, claves)

3. Service Discovery con Eureka
   - 3.1. Registrar y descubrir servicios automáticamente
   - 3.2. Comunicación con RestTemplate, FeignClient y WebClient usando nombres de servicio
   - 3.3. Configuración cliente-servidor
   - 3.4. Zona y cluster awareness

4. Spring Cloud Gateway (API Gateway)
   - 4.1. Enrutamiento inteligente basado en path, headers o query params
   - 4.2. Filtros pre/post personalizados
   - 4.3. Integración con JWT y autenticación centralizada
   - 4.4. Rate Limiting y Circuit Breaker con Resilience4j

5. Feign Clients y Load Balancing
   - 5.1. Declaración de clientes REST con interfaces (@FeignClient)
   - 5.2. Balanceo de carga con Spring Cloud LoadBalancer
   - 5.3. Manejo de errores y timeouts
   - 5.4. Fallbacks con Resilience4j

6. Tolerancia a Fallos con Resilience4j
   - 6.1. Circuit Breaker
   - 6.2. Retry
   - 6.3. RateLimiter

6.4. Bulkhead
6.5. TimeLimiter
6.6. Integración con Feign y RestTemplate

7. Observabilidad y Monitoring
   7.1. Spring Boot Actuator en microservicios
   7.2. Micrometer + Prometheus + Grafana
   7.3. Logs distribuidos con ELK (Elasticsearch, Logstash, Kibana)
   7.4. Correlación de trazas con Spring Cloud Sleuth y Zipkin

8. Seguridad Centralizada
   8.1. Autenticación y autorización con JWT
   8.2. OAuth2 y OpenID Connect con Spring Authorization Server o Keycloak
   8.3. Seguridad en Gateway (filtro de autenticación global)
   8.4. Seguridad por roles y scopes (@PreAuthorize, @Secured)

9. Comunicación Asíncrona y Mensajería
   9.1. RabbitMQ y Kafka con Spring Cloud Stream
   9.2. Productores y consumidores
   9.3. Event-driven architecture: @StreamListener, @SendTo, @KafkaListener
   9.4. Garantías de entrega y particionamiento
   9.5. Saga Pattern simplificado (Coreografía vs Orquestación)

10. Testing y Despliegue
    10.1. Tests de contrato con Spring Cloud Contract
    10.2. Pruebas de integración entre servicios con TestContainers
    10.3. Empaquetado y despliegue con Docker y Docker Compose
    10.4. CI/CD con GitHub Actions, GitLab CI o Jenkins
    10.5. Observabilidad y readiness/liveness probes para Kubernetes

RECOMENDABLE TENER CONOCIMIENTOS PREVIOS EN:

- Java y POO
- Fundamentos de Spring Boot
- Git / GitHub
- Gestión básica de base de datos con JPA/Hibernate
- REST y JSON
- Conceptos fundamentales de redes y sistemas

**PREGUNTAS FRECUENTES:**

- ¿Con qué base de datos se trabajará?
  Se trabajará con MySQL para los ejemplos

- ¿Qué debo conocer sobre Java exactamente?
  Programación orientada a objetos, pilares, interfaces, sobrecarga, manejo de excepciones, estructuras de datos, anotaciones.

- ¿Qué debo conocer sobre Spring Boot?
  Poder crear un proyecto con SpringBoot Initializr, configuraciones en application.properties,, poder usar anotaciones @RestController, @GetMapping, @RequestParam, @PathVariable, etc. Inyección de dependencias con @Autowired

- ¿Qué debo saber sobre Hibernate o JPA?
  Uso de anotaciones como @Entity, @Id, @GeneratedValue, consultas con JpaRepository, relaciones (@OneToMany, @ManyToOne, etc.)

- ¿qué debo saber sobre REST y JSON?
  Qué es API REST, cuáles son los verbos, diferencias de estado, estructura de JSON cómo serializar y deserializar.

- ¿A qué se refiere con Git y GitHub?
  Debe saber el manejo básico de Git como commits, push, pull y clonar

- ¿Algún conocimiento adicional ideal?
  Se valora conocer sobre Docker

- ¿Qué conocimientos de sistemas?
  Uso básico de línea de comandos del Sistema Operativo Windows e idealmente GNU/Linux.

- ¿Qué se necesita saber sobre redes?
  Conceptos básicos como IP, puerto o protocolo HTTP, saber iniciar y probar un servidor, entender qué es un endpoint y conocer diferencias entre entornos de desarrollo y producción.

- ¿Puedo seguir los ejemplos si no tengo Windows?
  Sí, se pueden seguir los ejemplos desde GNU/Linux, MacOS o Windows sin problemas, los programas son multiplataforma.

- ¿Nos entregarán los instaladores de los programas que se usen en clases?
  Entregaré enlaces de descarga y se mostrará su modo de descarga o instalación de ser necesario.

- ¿Será un curso completamente práctico?
  Sí, la teoría necesaria se la irá mostrando a la par de los ejemplos mientras se avanza en el curso.