## CURSO de ESPECIALIZACIÓN en:

# TECNOLOGIAS JAVA

| | | |
|---|---|---|
| **MODULOS** | JAVA STANDARD EDITION | 2 SEMANAS |
| | HIBERNATE | 2 SEMANAS |
| | SPRING BOOT | 2 SEMANAS |
| | SPRING BOOT Avanzado y Microservicios con SPRING CLOUD | 3 SEMANAS |
| | APACHE KAFKA & QUARKUS | 3 SEMANAS |
| **Requisitos** | Nociones de programación y bases de datos | |
| **Duración** | **3 MESES Lunes a Viernes** | |

## JAVA STANDARD EDITION

### Contenido Mínimo

1. Introducción a JAVA
2. Compilación y ejecución de programas
3. Tipos de datos
4. Estructura del lenguaje Java
   - 4.1. Secuenciales
   - 4.2. Declaración de variables
   - 4.3. Asignación de variables
   - 4.4. Condicionales
   - 4.5. Estructuras condicionales
   - 4.6. Estructuras iterativas
5. Casting de datos
6. Programación Orientada a Objetos
   - 6.1. Introducción a la Programación Orientada a Objetos
   - 6.2. Creación de clases
   - 6.3. Instanciación de objetos
   - 6.4. Declaración de propiedades y métodos
   - 6.5. Constructores
   - 6.6. Manejo de objetos
   - 6.7. Destructores
   - 6.8. Modificadores de acceso: public–private– protected
   - 6.9. Variables estáticas y finales.
   - 6.10. Métodos estáticos.
   - 6.11. Manejo de objetos predefinidos: Date – Integer – Float …
   - 6.12. Herencia
   - 6.13. Modificadores de acceso a clases: public – final - abstract
   - 6.14. Clases Abstractas
   - 6.15. Clases Finales
   - 6.16. Casting de clases

- 6.17. Interfaces
- 7. Excepciones y tratamiento de errores
- 8. JavaFX
  - 8.1. Manejo de Componentes de ventanas
    - 8.1.1. Label
    - 8.1.2. TextField
    - 8.1.3. Button
  - 8.2. CheckBox y otros
- 9. Eventos
- 10. Streams
  - 10.1. Tipos de streams
  - 10.2. Que es un file stream
  - 10.3. Que son los buffers
- 11. Conexión a Base de Datos con JDBC
  - 11.1. Introducción
  - 11.2. Consultas
  - 11.3. Inserción de datos
  - 11.4. Actualización de datos
  - 11.5. Eliminación de datos
  - 11.6. Transacciones

# HIBERNATE
## Contenido Mínimo

1. Introducción y Conceptos
   - 1.1. Introducción a los ORM
   - 1.2. Hibernate ORM
   - 1.3. Java Persistence API
2. Implementación DAO
   - 2.1. Instalación y Configuración
   - 2.2. Hibernate Dialectos
   - 2.3. Mapeo de Entidades
   - 2.4. La Clase HibernateUtil
   - 2.5. Hibernate usando XML
   - 2.6. Hibernate usando Anotaciones
   - 2.7. Ejemplo CRUD con Hibernate
3. Beans y mapeamiento hibernate básico
   - 3.1. Mapeo hibernate Unidireccional
   - 3.2. Mapeo hibernate Bidireccional
   - 3.3. Mapeo hibernate con Archivos
   - 3.4. Mapeo hibernate con Anotaciones
   - 3.5. Mapeo hibernate relaciones uno a uno
   - 3.6. Mapeo hibernate relaciones uno a muchos
   - 3.7. Mapeo hibernate relaciones muchos a muchos
4. Hibernate Query Languaje – HQL
   - 4.1. Consultas HQL

4.2. Objetos y Colecciones
4.3. Paso de Parámetros
4.4. Optimizando Consultas
4.5. Consultas Nativas

# SPRING BOOT
## (Fundamentos + REST API + BD + Seguridad)

1. Introducción a Spring Boot
    1.1 ¿Qué es Spring Boot? (Ventajas sobre Spring tradicional)
    1.2 Creación de proyectos con:
        - Spring Initializr
        - Spring Boot CLI (opcional)
    1.3 Estructura de un proyecto Spring Boot
    1.4 Configuración básica:
        - application.properties vs application.yml
        - Cambiar puerto y otros ajustes
2. Controladores y REST API
    2.1 Diferencias entre @Controller y @RestController
    2.2 Mapeo de endpoints:
        - @RequestMapping, @GetMapping, @PostMapping, etc.
        - @PathVariable y @RequestParam
    2.3 Manejo de respuestas:
        - ResponseEntity
        - DTOs (Data Transfer Objects)
    2.4 Validación de requests:
        - @Valid, @NotNull, @Size, etc.
3. Inyección de Dependencias y Spring IOC
    3.1 @Component, @Service, @Repository
    3.2 Inyección por constructor (best practice) vs @Autowired
    3.3 Ciclo de vida de los Beans (@PostConstruct, @PreDestroy)
4. Conexión con Bases de Datos (JPA/Hibernate)
    4.1 Configuración de spring-data-jpa
    4.2 Entidades y anotaciones (@Entity, @Id, @OneToMany, etc.)
    4.3 Repositorios (JpaRepository)
    4.4 Consultas personalizadas (@Query)
    4.5 Transacciones (@Transactional)
5. Manejo de Excepciones
    5.1 @ControllerAdvice y @ExceptionHandler
    5.2 Respuestas de error personalizadas
6. Seguridad con Spring Security
    6.1 Autenticación básica (HTTP Basic)

6.2 JWT (JSON Web Tokens)
6.3 Configuración de roles y permisos (@PreAuthorize)
7. Otras Características Importantes
7.1 Logging con SLF4J/Logback
7.2 Pruebas unitarias (@SpringBootTest, MockMvc)
7.3 Internacionalización (i18n)
7.4 CORS (Configuración global y por endpoint)
8. Despliegue
8.1 Generar un JAR ejecutable
8.2 Desplegar en Tomcat

# SPRING BOOT Avanzado y Microservicios con SPRING CLOUD

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

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 (mensajería y eventos)
   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
   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. Arquitectura basada en eventos aplicada a microservicios
   9.2. Spring Cloud Stream y modelo de abstracción
   9.3. Productores y consumidores a nivel framework
   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

# APACHE KAFKA & QUARKUS
## (FRAMEWORKS DE APACHE)

### Contenido Mínimo

1. **Fundamentos de Apache Kafka**
2. Instalación en máquina local
   2.1. Instalación de Kafka
   2.2. Instalación de Zookeeper
   2.3. Configuración de clúster

3. Productores y consumidores
    3.1. Conceptos
    3.2. Creación de productores y consumidores
    3.3. Configuraciones
4. Temas y particiones
    4.1. Gestión de temas y particiones
    4.2. Comandos de administración
5. Monitoreo y mantenimiento
    5.1. Prometheus y Grafana
    5.2. Monitoreo básico en un clúster
    5.3. Mantenimiento y reducción de problemas
6. **Introducción a Quarkus**
    6.1. Diferencias con Spring Boot y otros frameworks
    6.2. Instalación y configuración inicial
7. Desarrollo de aplicaciones con Quarkus
    7.1. Creación de un proyecto Quarkus desde cero
    7.2. Uso de Quarkus Dev Mode (Live Coding)
    7.3. Configuración y uso de Quarkus con REST y JSON
8. Persistencia y Base de Datos
    8.1. Introducción a Hibernate ORM con Panache
    8.2. Configuración de Datasources y conexión con PostgreSQL
    8.3. Uso de Repositorios y consultas reactivas
9. Despliegue y optimización
    9.1. Creación de imágenes nativas con GraalVM
    9.2. Despliegue en Kubernetes y contenedores