# PARCIAL 2-MICROSERVICIOS

**Objetivo del Parcial:**

Desarrollar una continuación del proyecto presentado en el archivo base, aplicando los conocimientos sobre microservicios con Spring Boot y Spring Cloud, en un entorno distribuido, guiado paso a paso por cada tema visto en clase. El objetivo es transformar parte del sistema en microservicios independientes, documentados y desplegados.

## 1. Configuración del Proyecto General

**Paso a paso:**

1. Crea una organización o repositorio en GitHub con el nombre: sistema-educativo-microservicios-nombre
2. Crea un archivo README.md con una descripción del sistema y el enfoque distribuido.
3. Dentro del repositorio, crea carpetas por microservicio: usuarios-servicio, asignaturas-servicio, entre otros.

**Complementación de los estudiantes:** Incluir datos personales, estructura del repositorio y visión general del sistema.

## 2. Desarrollo de Microservicios con Spring Boot

**Paso a paso:**

1. Crear al menos 3 microservicios:
   - usuarios-servicio: gestión de estudiantes y docentes.
   - asignaturas-servicio: CRUD de materias.
   - matriculas-servicio: registro de estudiantes en materias.
2. Cada microservicio debe tener su propio application.properties y base de datos.

**Complementación del estudiante:** Implementar controladores, servicios y entidades.

## 3. Comunicación entre Microservicios

**Paso a paso:**

1. Usar Feign Client en el matriculas-servicio para consumir datos de usuarios-servicio y asignaturas-servicio.
2. Simular una matrícula completa.

**Complementación del estudiante:** Crear cliente Feign y probar integración**.**

**4. Gestión de Configuración y Descubrimiento de Servicios**

**Paso a paso:**

1. Implementar un config-server y un repositorio de configuración en GitHub.
2. Crear un eureka-server y registrar todos los microservicios.

**Complementación del estudiante:** Documentar cómo se conectan y su configuración.

**5. Seguridad en Microservicios (JWT)**

**Paso a paso:**

1. Implementar autenticación en el usuarios-servicio.
2. Proteger los endpoints con Spring Security y JWT.

**Complementación del estudiante:** Generar y validar tokens, definir roles.

**6. Monitorización y Registro**

**Paso a paso:**

1. Agregar Spring Boot Actuator a los microservicios.
2. Exponer endpoints de salud y métricas (/actuator/health).

**Complementación del estudiante:** Crear dashboard o consola de monitoreo.

**7. Pruebas de Microservicios**

**Paso a paso:**

1. Crear pruebas unitarias y de integración con Spring Boot Test y Postman.
2. Validar comportamiento de controladores y servicios.

**Complementación del estudiante:** Incluir mínimo 1 test por microservicio.

**8. Despliegue y Orquestación**

**Paso a paso:**

1. Crear un archivo Dockerfile por microservicio.
2. Crear un docker-compose.yml para levantar todo el entorno.

**Complementación del estudiante:** Ejecutar localmente y registrar evidencia en el README.

## 9. Entrega Final

El estudiante debe entregar:

- Repositorio en GitHub con el código completo.
- Documento PDF con capturas de pantalla, explicaciones breves de cada paso, arquitectura general y aprendizajes.
- Enlace público del repositorio: https://github.com/...

### Criterios de Evaluación

| Criterio | Valor (%) |
|---|---|
| Estructura y organización del repositorio | 10% |
| Implementación funcional de microservicios | 25% |
| Comunicación entre microservicios | 10% |
| Configuración y descubrimiento de servicios | 10% |
| Seguridad (JWT y control de accesos) | 10% |
| Pruebas unitarias e integración | 10% |
| Monitorización y documentación técnica | 10% |
| Despliegue con Docker | 10% |
| Presentación clara del Word explicativo | 5% |

### Rúbrica

| Nivel de desempeño | Descripción |
|---|---|
| Excelente (5) | Cumple con todos los requisitos, código limpio, pruebas funcionales, despliegue exitoso, documentación clara. |
| Bueno (4) | Cumple con casi todos los requisitos, pequeños errores o ausencias menores, documentación aceptable. |
| Aceptable (3) | Cumple con los aspectos básicos, faltan pruebas o documentación, errores leves en el funcionamiento. |
| Bajo (2) | Presenta varios errores técnicos, documentación deficiente, pruebas ausentes, dificultad para explicar su trabajo. |
| Insuficiente (1) | No cumple con los requerimientos mínimos, estructura incompleta, código no funcional, sin evidencias ni explicación clara. |

**¡Mucho ánimo! Este parcial es una oportunidad para demostrar tu dominio en microservicios aplicados a un sistema real.**