**Guía de Evaluación Parcial 2 - DSY1103 Desarrollo Full Stack I**

**Objetivo General de la Evaluación Parcial 2**

Esta evaluación corresponde a la segunda fase del desarrollo del caso semestral. A partir de la propuesta arquitectónica desarrollada en la Evaluación Parcial 1, los estudiantes deben **configurar e implementar microservicios funcionales** conectados a una **base de datos MySQL**, desarrollando **operaciones CRUD** completas y gestionando el proyecto bajo buenas prácticas de programación.

Esta etapa está centrada en los Resultados de Aprendizaje **RA2** (configuración de proyectos) y **RA3** (desarrollo de microservicios conectados entre sí).

**✅ Qué deben entregar y desarrollar**

### **1. Desarrollo funcional (programación)**

* Crear uno o más microservicios definidos en la propuesta de arquitectura.
* Implementar el modelo de datos con **entidades JPA** (Java + Hibernate).
* Configurar el acceso a **base de datos MySQL** mediante application.properties.
* Crear los paquetes model, repository, service, controller.
* Programar operaciones CRUD usando JpaRepository y anotaciones como @RestController, @GetMapping, @PostMapping, etc.
* Implementar la **comunicación RESTful** entre microservicios (por ejemplo, mediante llamadas HTTP entre servicios).
* Validar y demostrar el funcionamiento de los endpoints utilizando **Postman**.

### **2. Informe (entrega escrita)**

El informe debe incluir los siguientes apartados:

#### **A. Desarrollo de Microservicios**

* Descripción de los microservicios desarrollados.
* Diagrama de arquitectura actualizada (puede reutilizar el de la Parcial 1, si se ha ajustado).
* Capturas de código relevantes (endpoints, servicios, repositorios).
* Capturas de pruebas funcionales en Postman.

#### 

#### **B. Base de Datos**

* Modelo entidad-relación implementado.
* Capturas de las clases de entidades Java.
* Capturas de la configuración de application.properties y de la conexión exitosa a la base de datos.

#### **C. Integración de Servicios**

* Descripción teórica del flujo de trabajo CI/CD (a nivel conceptual, según lo explicado por el docente).
* Explicación del entorno de prueba: pruebas con Postman y, opcionalmente, interfaz de apoyo desarrollada por el docente para mostrar respuestas JSON.

#### **D. Pruebas Unitarias**

* Implementación de al menos una prueba unitaria con JUnit para un servicio o componente simple.
* Documentación breve de resultados (captura o descripción).

#### **E. Buenas prácticas y colaboración**

* Captura del flujo de trabajo con Git (commits, ramas, integraciones).
* Breve explicación del uso colaborativo del repositorio.
* Ejemplos de buenas prácticas aplicadas (nombres claros, modularidad, uso de anotaciones, REST).

**🏛️ Presentación (evaluación individual)**

Cada estudiante debe presentar:

1. El diseño y estructura del proyecto (carpetas, clases, flujo general).
2. La configuración de pom.xml, dependencias y uso de Maven.
3. Ejemplos de uso de Git y colaboración efectiva.
4. Un ejemplo completo de CRUD con pruebas reales en Postman.
5. Explicación de las relaciones entre entidades (por ejemplo, @OneToMany).
6. Una explicación clara de cómo se integran los microservicios REST entre sí.
7. Un ejemplo de cómo una respuesta JSON podría usarse en un frontend (solo como idea ilustrativa, no se desarrolla frontend).
8. Una reflexión sobre cómo mejoraron desde la Parcial 1.

**🔄 Formato sugerido de entrega (informe)**

1. Portada
2. Tabla de contenido
3. Desarrollo de microservicios (con capturas)
4. Base de datos y configuración
5. Integración de servicios (descripción y pruebas)
6. Pruebas unitarias
7. Buenas prácticas y colaboración (Git)
8. Conclusión y reflexión
9. Anexos (enlaces a repositorio, video de prueba si aplica)

**Importante para docentes**: el foco de esta evaluación está en lo **backend y la integración funcional**, no en el diseño del frontend. La documentación, la modularidad del código y el uso efectivo de Git son aspectos claves de la evaluación. Las pruebas se validarán preferentemente con Postman y la presentación oral permite verificar el dominio individual del estudiante.